Bolt Beranek and Newman Inc.

REPORT NO. 4159

SPEECH COMPRESSION AND SYNTHESIS

ANNUAL REPORT
AUGUST 1979

PREPARED FOR:
ADVANCED RESEARCH PROJECTS AGENCY

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER BBN Report No. 4159	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER 9
4. TITLE (and Subtitle) 6 SPEECH COMPRESSION AND SYNTHESIS.		5. TYPE OF REPORT & PERIOD COVERED Annual Rep[redacted]t 6 Apr[redacted] 78 — 7 Jun[redacted] 79
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) 10 Michael Berouti, Lynn Cosell, John Klovstad, John/Makhoul, Richard/Schwartz, Jared Wolf		8. CONTRACT OR GRANT NUMBER(s) 15 F19628-78-C-0136 ✓ARPA Order-3515
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 50 Moulton Street Cambridge, Massachusetts		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS 12 114p		12. REPORT DATE 11 Aug[redacted]79
		13. NUMBER OF PAGES 111
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Deputy for Electronic Technology (RADC/ETC) Hanscom Air Force Base, MA 01731 Contract Monitor: Mr. Caldwell P. Smith		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce, for sale to the general public.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

14 BBN-4159

18. SUPPLEMENTARY NOTES

This research was supported by the Defense Advanced Research Projects Agency under ARPA Order No. 3515.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Speech synthesis, phonetic synthesis, diphone, LPC synthesis, vocoder, speech compression, linear prediction, voice-excited coder, high-frequency regeneration, spectral duplication, real-time vocoder, mixed-source excitation model, adaptive lattice.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes our work for the past year on speech compression and synthesis. We implemented a real-time variable-frame-rate LPC vocoder operating at an average rate of 2000 bits/s. We also tested our mixed-source model as part of the vocoder. To improve the reliability of the extraction of LPC parameters, we implemented and tested a range of adaptive lattice and autocorrelation algorithms. For data rates above 5000 bits/s, we developed and tested a new high-frequency regeneration technique, spectral duplication,

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

(over)

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

060 100

20. Abstract (cont)

which reduces the roughness in the synthetic speech. As part of an overall very-low-rate speech compression system, we designed and implemented a program for phonetic synthesis from diphone templates. The lengthy process of extracting the 3000 needed diphone templates has been half completed.

SPEECH COMPRESSION AND SYNTHESIS

Annual Technical Progress Report

6 April 1978 - 7 June 1979

ARPA Order No. 3515

Name of Contractor:
Bolt Beranek and Newman Inc.

Effective Date of Contract:
6 April 1978

Contract Expiration Date:
7 July 1980

Contract No. F19628-78-C-0136

Principal Investigators:
Dr. John Makhoul
(617) 491-1850, x4332

Dr. R. Viswanathan
(617) 491-1850, x4336

Sponsored by
Defense Advanced Research Projects Agency (DoD)
Monitored by RADC/ETC

PROJECT PERSONNEL

John Makhoul	Supervisory Scientist
A.W.F. (Bill) Huggins	Senior Scientist
R. (Vishu) Viswanathan	Senior Scientist
Jared Wolf	Senior Scientist
Michael Berouti	Scientist
Lynn Cosell	Scientist
John Klovstad	Scientist
Richard Schwartz	Scientist
Dennis Klatt	Senior Consultant
Victor Zue	Senior Consultant
Nancy Chapman	Project Secretary

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

List of Figures

1. INTRODUCTION

In this annual report we present our work performed during the period April 6, 1978 to June 7, 1979 in the area of speech compression and synthesis.

In Section 1.1 we give a very brief list of the major accomplishments in the past year. The reader is referred to the body of the report for details on these as well as other accomplishments. An outline of this report is given in Section 1.2. In Section 1.3, we give a list of the presentations and publications for the past year. The publications are included in the Appendix.

1.1 Major Accomplishments

a) Developed RTFUD, an RT-11/FORTRAN debugging environment for the AP-120B vocoder program.

b) Brought up the ISI LPC-II real-time vocoder on our system. Modified the vocoder to include the optimum-linear-fit variable-frame-rate (VFR) algorithm and the mixed-source model, both developed at BBN.

c) Implemented and tested a range of adaptive autocorrelation and lattice algorithms for LPC analysis, which have implications for new hardware developments (such as switched-capacitor designs).

d) Developed a new high-frequency regeneration (HFR) technique, spectral duplication, to be used in baseband coders in the range above 5000 bits/s.

e) Designed and implemented a program for phonetic synthesis from diphone templates. The program was tested successfully using an initial set of 200 diphones. Designed and recorded a data base comprising a complete set of diphones (about 3000) for English. Began the lengthy process of extracting the diphone templates; we have completed the extraction of 1500 diphones.

1.2 Outline

In Section 2 we present our work towards bringing up a real-time vocoder and improving its output speech quality. Section 3 describes the results of new techniques for extraction and coding of linear prediction (LPC) parameters. A new high-frequency regeneration (HFR) technique is presented in Section 4 as a source model to be used with baseband coders. Finally, in Section 5, we present our progress in developing the phonetic synthesis component of a very low rate (VLR) coder.

1.3 Presentations and Publications

During the past year, we gave a number of oral presentations at the regular ARPA Network Speech Compression (NSC) Meetings. In addition, we made four presentations at conferences and had two papers published. These were:

a) J. Makhoul, "A Class of All-Zero Lattice Digital Filters: Properties and Applications," IEEE Trans. Acoustics, Speech and Signal Processing, pp. 304-314, Aug. 1978.

b) J. Makhoul, R. Viswanathan, R. Schwartz, and A.W.F. Huggins, "A Mixed-Source Model for Speech Compression and Synthesis," J. Acoust. Soc. Am., Vol.64, No.6, pp. 1577-1581, Dec. 1978.

c) J. Makhoul and M. Beyrouti, "Predictive and Residual Encoding of Speech," invited paper, J. Acoust. Soc. Am., Vol.64, Supplement No.1, paper YY2, p. S128, Fall 1978.

d) A.W.F. Huggins, R.M. Schwartz, R. Viswanathan, and J. Makhoul, "Subjective Quality Testing of a New Source Model of LPC Vocoders," J. Acoust. Soc. Am., Vol.64, Supplement No.1, paper GGG12, p. S161, Fall 1978.

e) J. Makhoul and M. Beyrouti, "High-Frequency Regeneration in Speech Coding Systems," IEEE Int. Conf. Acoustics, Speech, and Signal Processing, Washington, DC, pp. 428-431, April 1979.

f) R. Schwartz, J. Klovstad, J. Makhoul, D. Klatt, and V. Zue, "Diphone Synthesis for Phonetic Vocoding," IEEE Int. Conf. Acoustics, Speech, and Signal Processing, Washington, DC, pp. 891-894, April 1979.

Copies of papers a), b), e), and f), are given in the Appendix.

2. REAL TIME VOCODER

Much of our effort of the past year has been devoted to bringing up, modifying, and evaluating a real-time ARPANET vocoder. The original vocoder had been implemented at the Information Sciences Institute (ISI), using an FPS-120B Array Processor to perform the numerical calculations and a PDP-11 to transmit and receive coded speech across the network. The vocoder required a special operating system, EPOS, also designed and implemented at ISI, to handle the network transactions.

2.1 Installation

Installing the vocoder at BBN required several steps. We first had to obtain the necessary hardware and software and then configure both to our particular needs.

2.1.1 Operating System Installation

The process of installing EPOS on our PDP-11 was complicated by the fact that our PDP-11 was quite different from the one used at ISI. They used an 11/45, while we have an 11/40. The major differences are in the effective address spaces available on the two machines. ISI streamlined their code as much as possible to fit into our address space, and both sites spent a good deal of time debugging the BBN configuration of EPOS.

2.1.2 Hardware Installation

In the process of debugging EPOS, we discovered that we did not have enough physical memory on our PDP-11. We purchased and installed an additional 64K words of MOS , bringing the total memory on the PDP-11 to 128K.

In January 1978, Floating Point Systems, Inc. installed an AP-120B array processor on our PDP-11. We installed the AP program development software and library under RT11, and tested and accepted the system.

2.1.3 Program Development Software Installation

The program development software required to install the vocoder program includes the AP assembler, APAL, and the AP linker, APLINK. An unanticipated difficulty arose over this software. ISI performs all assembling and linking on their TENEX system, and our PDP-11 system would not accept the formats of the source files used by ISI. In addition, they had modified some of the FPS support software, which also ran under TENEX. We were forced to modify all of this support software to run under the TOPS-20 operating system.

2.1.4 Vocoder Program Installation

The successful modification of the support software enabled us to make several changes to the vocoder program. These changes to the vocoder program were required to accommodate differences between the ISI and BBN systems. The most significant difference is that ISI uses A/D and D/A converters interfaced directly to the AP-120B Array Processor, while BBN's converters are interfaced to the SPS-41. ISI had initially used the SPS-41 for analog/digital conversion, but for later versions of the vocoder, they switched to the AP-120B. This switch required identifying and changing various routines that use the converters. EPOS modifications were also necessary to allow access to the SPS-41.

2.1.5 Initial Vocoder Experiments

Using FUD, an ISI-developed debugger for the AP-120B, we tested the AP-120B coder for the vocoder. This test did not use the ARPA Network, but merely fed the parameters obtained from the analysis portion of the AP-120B program back to the synthesis portion of the program. We obtained moderately intelligible output speech in real-time. We were also able to store analysis parameters on a PDP-11 data file and subsequently transmit that file to ISI, who performed the synthesis. Using this testing procedure, we found and fixed bugs in the converter routines. We were able to obtain good quality speech using the vocoder in this "back-to-back" manner. We further tested the vocoder system by carrying on a conversation with ISI over the ARPA Network, and we also used the "echo" facility of the vocoder program to listen to ourselves over the network. This testing procedure pointed out a potentially serious problem. The PDP-11/40 seemed unable to keep up with the real-time speech requirements. Because ISI uses a PDP-11/45, a significantly faster machine, they had not encountered this problem. We were able to change the priorities of certain tasks in the PDP-11 portion of the vocoder, and to speed up the coder somewhat, so that we no longer have timing problems in point-to-point conversations. However, there are still real-time problems that surface during conferencing.

The network experiments also demonstrated that the speech quality was considerably worse over the network than that obtained from the "back-to-back" system. The problem was determined to be in BBN's receiver. With the help of ISI, we were able to find and fix a subtle bug in the real-time buffering code. This bug resulted from the configuration differences.

We also participated in several conferencing experiments with Lincoln Laboratories and ISI. Unfortunately, due to the lack of speed in the 11/40 mentioned above, we were unable to communicate with both of the other participants. We are currently working with ISI to find modifications that will support conferencing on our configuration.

2.2 Vocoder Modifications

Once we had established the "current" ISI version of the vocoder at BBN, we were able to modify it to include some of the algorithmic recommendations we had made previously. At the same time, we attempted to make the vocoder more modular, and to keep our modifications as modular as possible.

2.2.1 Support Software

The difficulty we experienced in debugging the initial vocoder convinced us that more effective debugging and testing tools were necessary. We also found that, while it was fine for real-time operation, EPOS had serious drawbacks as a debugging environment. The most effective debugging could be done if the vocoder were running in the Fortran environment under RT-11.

2.2.1.1 APEX and APLINK

The most general system for using the AP-12ØB for array processing is the AP Executive (APEX) supplied by Floating Point Systems Inc. APEX allows a host (PDP-11) FORTRAN program to transfer data to and from the AP-12ØB and to specify computational operations to be performed on the data by the AP-12ØB. The APEX library (APLIB) includes an extensive repertoire of elementary computational operations, and the user may program his own functions, to be used in the same way.

The APEX system, as delivered by FPS, is written in FORTRAN, which on our PDP-11 operating system (RT-11) produces inefficient code. As a result, each APEX routine called from the Host FORTRAN program consumes 2 to 4 ms of PDP-11 time just to start the computation process in the AP-120B. Many of these elemental operations require on the order of 1 ms to run in the AP-120B, so most of the time for running a multicall APEX program is spent in PDP-11 APEX overhead.

Analysis of the operations of APEX in starting an AP-120B computation showed that efficiency could be gained from (1) recoding the time-critical sections of APEX in machine language to avoid inefficient FORTRAN code and to eliminate the time-consuming nesting of FORTRAN subroutine calls and (2) streamlining the manner in which APEX is called by each individual user-called routine, once again eliminating needless nesting of FORTRAN subroutine calls. The latter was accomplished by modifying APLINK, one of the AP-120B program development programs. The resulting modified versions of these programs, APEX2 and APLINK2, were then tested, and the overhead associated with an APEX call was found to have been reduced to 0.5 to 1.0 ms.

2.2.1.2 SPS-FPS Support

We developed and tested a package of PDP-11 subroutines that control transfers between the SPS-41 converters and the AP-120B Main Data memory. These subroutines automatically load SPS-41 program memory, initialize the sampling rate counters, and manage the double buffering required for real-time operation. They also return input and output data magnitude information to the calling program and indicate whether the real-time constraints have been met, and, if not, how much additional time was required. These subroutines have been designed to operate under APEX, the FPS-supplied executive.

2.2.1.3 IMSYS

Graphic displays are an invaluable aid to signal processing research. Since 1971, we have been using an IMLAC PDS-1 display computer in conjunction with our PDP-10 systems for interactive graphics support of our speech and signal processing research. A major part of the support software for the use of this graphics system has been IMSYS, a general-purpose graphics control program for the IMLAC, which creates and maintains displays specified by user programs operating in the host PDP-10. IMSYS graphics packages that are usable by user programs written in INTERLISP, FORTRAN, and BCPL have put interactive graphics at the disposal of a wide variety of user programs.

In order to use this graphics facility in our testing procedures, we extended the usability of IMSYS graphics to FORTRAN programs running on our Speech Processing PDP-11/40. The IMLAC part of IMSYS is used without modification. The host IMSYS graphics package was taken largely intact from another BBN project that had implemented IMSYS in RSX-11D FORTRAN; only new host-to-IMLAC data exchange routines and minor housekeeping changes were necessary.

2.2.1.4 RTFUD

With the preliminary support software developed, we were able to implement RTFUD, a RT-11/FORTRAN/debugging environment for the AP-120B vocoder program. RTFUD operates the vocoder in non-real-time, using files for input and output of digitized speech waveforms and/or transmission parameters. Non-real-time operation implies that the vocoder can be stopped, examined, and then continued. Furthermore, the implementation in FORTRAN has permitted the rapid implementation of a number of statistics-gathering and diagnostic tools. RTFUD is used with AP FDT, a FORTRAN debugging program (FDT) that we have modified to permit it to debug the AP-120B as well as the PDP-11 program. The facilities of RTFUD have uncovered several bugs and problems in the AP-120B vocoder program, and they have been used for developing the variable frame rate (VFR) and synthesizer modifications described below. This section gives a brief description of the facilities offered by the current implementation of RTFUD.

Figure 2.1 illustrates the relation of the real-time AP-120B program to the RTFUD implementation. In the real-time implementation, the AP-120B vocoder program is organized as a closed loop. In RTFUD, the analysis (XANAL) and synthesis (XSYNTH) portions of the program are separately and explicitly controlled by the RTFUD program (by means of APEX, the FPS-supplied "AP-120B Executive") in the PDP-11. The analysis and synthesis portions of the vocoder programs are effectively identical in both cases. Therefore, developments made to the AP-120B programs in the non-real-time configuration are directly transferable to the real-time system; RTFUD does not compromise the real-time goals of the vocoder development project.

The basic mode of operation of the RTFUD vocoder is "back-to-back", with digitized speech input from a file processed by the analysis and synthesis portions of the vocoder, producing synthesized speech output to a file. RTFUD also provides a real-time playback facility, which converts a disk file to speech for listening purposes. RTFUD also provides for "analysis" and "synthesis" modes of operations, in which the respective output or input is a file containing transmission data (which in an actual real-time system would be transmitted through the communications network).

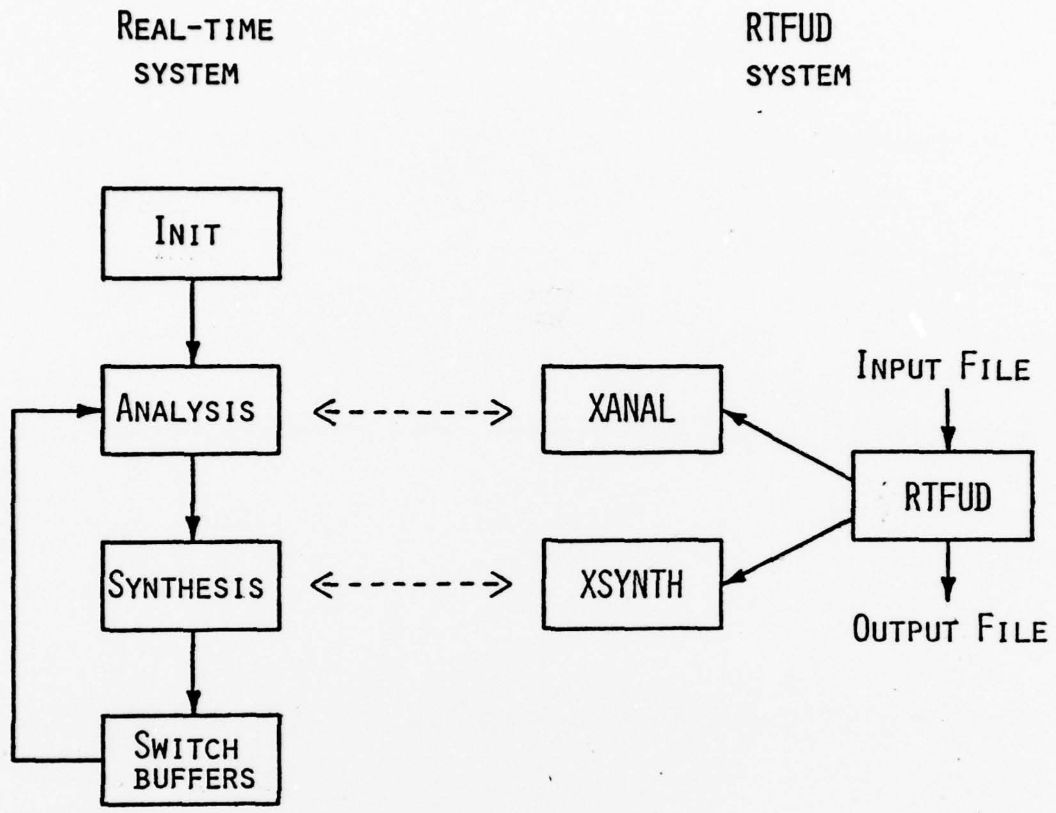


Fig. 2.1 Comparison of the AP-120B vocoder program in the real-time system and in the RTFUD system.

Figure 2.2 illustrates the value of being able to use a known, repeatable input to the vocoder and being able to observe the input and output waveforms in detail. The figure shows three versions of the waveform of the syllable "mass" from the word "Massachusetts". The top trace is the original digitized speech; the center trace shows the same syllable at the output of the ISI vocoder. A

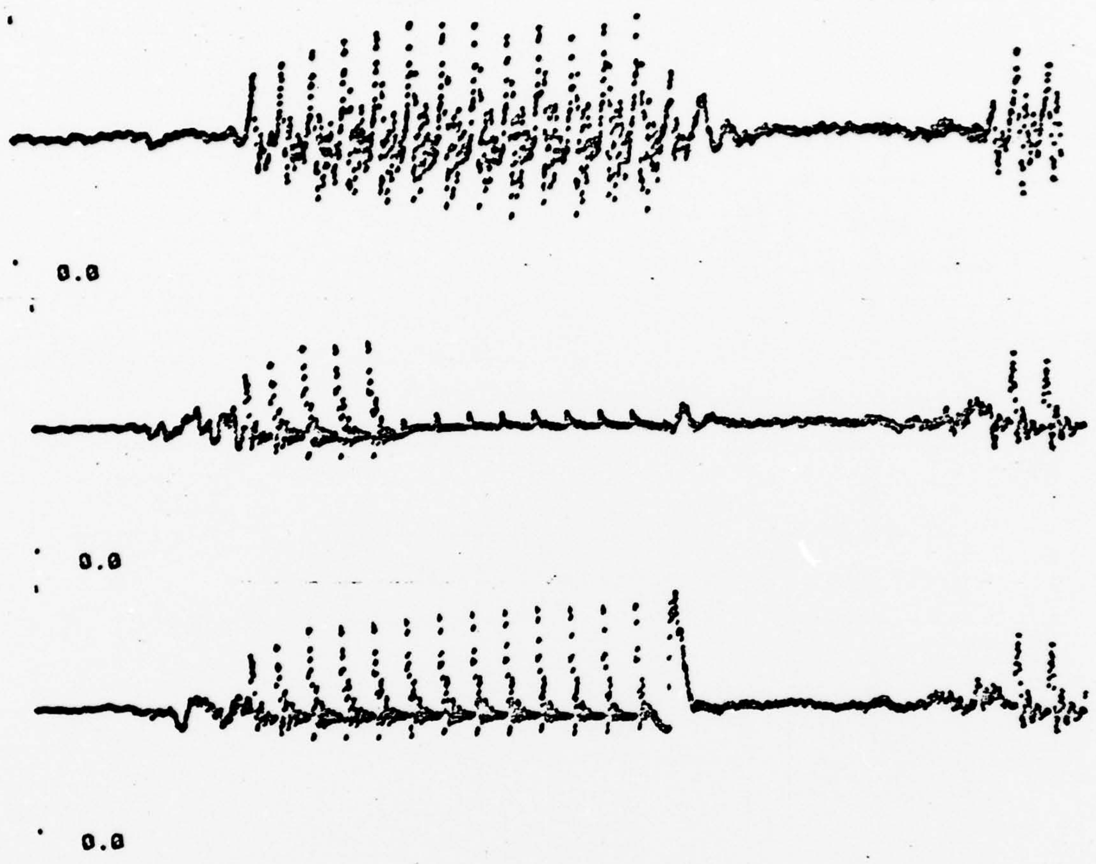


Fig. 2.2 Waveforms of the syllable "mass" in "Massachusetts": (top) original; (center) vocoded with gain bug; (bottom) vocoded with gain bug fixed.

program bug has inappropriately lowered the gain about half-way through the syllable. By editing the input file to contain just this syllable, and observing certain vocoder memory locations after each frame of speech, it was an easy task to localize and correct this bug in the synthesis interpolation routine. The output speech after this bug was fixed is shown in the bottom trace of Figure 2.2.

RTFUD accumulates several statistics of the vocoder, which can be valuable in understanding and adjusting the operation of the vocoder. Figure 2.3 illustrates the type of statistics currently printed.

```
>BACK-TO-BACK
 INPUT SPEECH FILE: DK1:SC2A.WAV
    54501. SAMPLES @    150  USEC =        8.175 SEC
 OUTPUT SPEECH FILE: *DK1:SC2AB.WAV

   872 FRAMES
 PDP-11 TIME =    35.9 MS/FRAME
 AP-120B TIME (ANALYSIS)  =   3.4 MS/FRAME
 AP-120B TIME (SYNTHESIS) =   2.0 MS/FRAME
 ANALYSIS:  27.8 BITS/FRAME, 2895. BITS/SEC
 SYNTHESIS: 27.8 BITS/FRAME, 2895. BITS/SEC
 PITCH, GAIN, K'S:  44.4  51.2  57.2 FRAMES/SEC (OUT OF 104.2)
```

Fig. 2.3 Statistics summary output by RTFUD.

RTFUD contains provisions for accumulating histogram data on the transmitted parameters. Figure 2.4 shows a histogram display derived from vocoding about 10 sec of speech from one (male) speaker. (The histogram display was made with IMSYS, the graphics system implemented on our PDP-11.) Several valuable observations about the vocoder operation may be made from this figure. Note

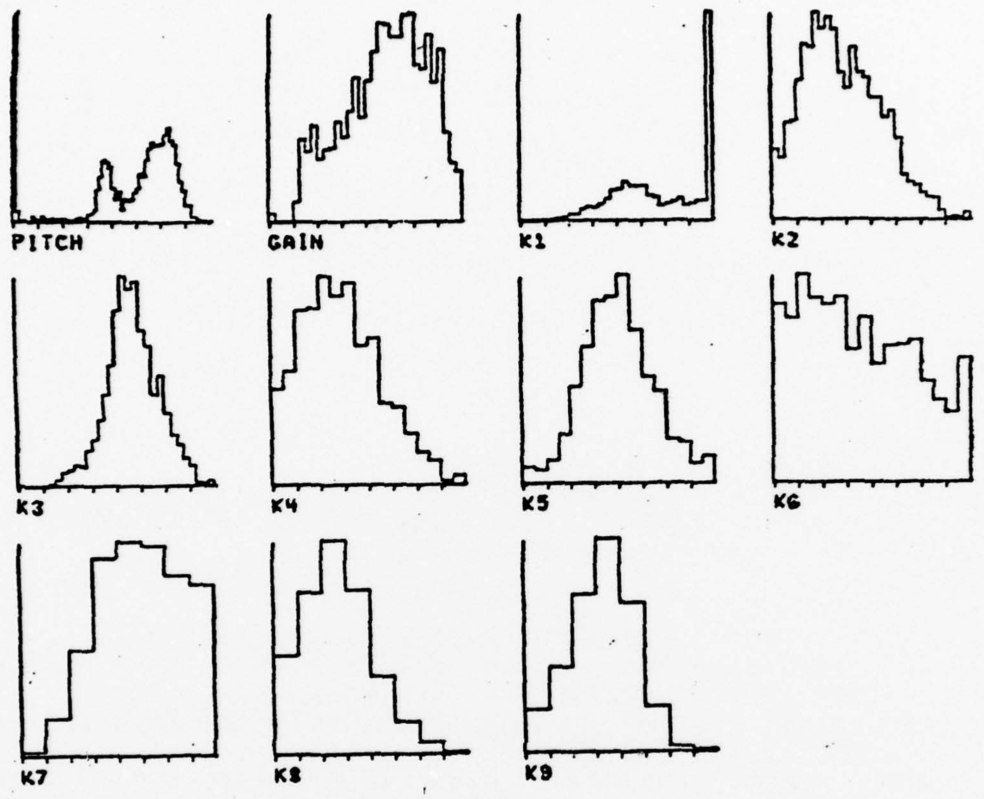


Fig. 2.4 Histogram display of transmitted parameters, single (male) speaker.

that the pitch period histogram is distinctly bimodal. The right-hand lump represents correct pitch periods for this speaker, and the left-hand lump represents octave errors (pitch periods of half the correct period). Also note the K1 (first reflection coefficient) histogram. It shows that the highest coded value of

K1 is used a disproportionate amount of the time. In other words, the quantization tables appear not to encompass the full range of K1 computed by the analysis portion of the vocoder. Figure 2.5

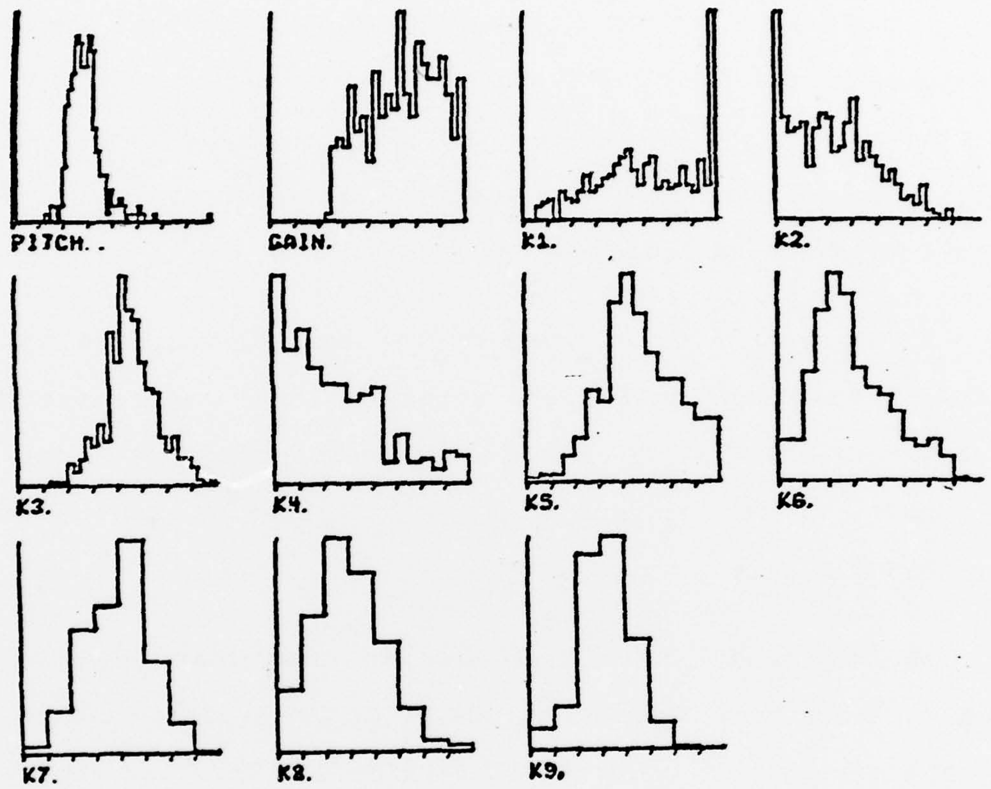


Fig. 2.5 Histogram display of transmitted parameters, single (female) speaker.

shows a comparable histogram for another (female) speaker. The pitch period histogram shows shorter pitch periods, as expected, and few (if any) octave errors. The K1 histogram shows the same effect as before, and for this speaker, K2 and K4 are also severely skewed.

RTFUD contains provisions for producing frame-by-frame listings of various vocoder parameters such as analysis parcels (before VFR decisions), synthesis parcels (after VFR decisions), and unquantized reflection coefficients. Because the FORTRAN environment of RTFUD facilitates such formatted output, during the testing of the new VFR algorithm it was a simple matter to add a special printout of the VFR tables so that the operation of the VFR algorithm could be monitored and verified.

RTFUD also contains commands for controlling such things as enabling or disabling VFR operation of the vocoder, controlling double-threshold pitch and gain transmission (not previously implemented), and adjusting the VFR thresholds for the desired transmission rate.

We have made RTFUD available to the other Network Speech Compression sites. Currently, ISI is in the process of bringing it up on their system, under RT-11 version 3. They are experiencing some compatibility problems between version 3 and version 2, under which we developed RTFUD.

2.2.2 Lattice Synthesis Modification

We modified the vocoder to use a lattice-form synthesis filter instead of the normalized form that was previously used. The normalized-form filter was advantageous in the SPS-41 vocoder implementation, but not in the AP-120B. The lattice-form requires only two multiplies per stage, compared to four for the normalized-form. We used the computation time saved by this reduction to make the synthesis routine more modular. Where before, the arguments of the routine were hand compiled and were an integral part of the design of the vocoder, they are now passed to the subroutine and can be general.

The normalized-form filter required the arc-sine of the reflection coefficients, used as pointers to the filter coefficients. Decoding of transmitted parameters and linear interpolation were performed in this arc-sine domain. The lattice-form filter uses the reflection coefficients themselves. However, since the new VFR algorithm assumes linear interpolation in the LAR domain (for reasons of spectral sensitivity), we modified the vocoder synthesizer to decode and interpolate in this domain. We then added a subroutine to convert from LARs to reflection coefficients for the synthesis filter computations.

2.2.3 New Variable Frame-Rate Modification

We implemented and tested the new VFR algorithm described in [1]. We have successfully modified the vocoder to use this new algorithm.

The new VFR algorithm operates by modeling what would occur in the synthesis process if the current packet were transmitted. If the average difference between the actual LAR values since the last transmitted frame and the interpolated values is less than a threshold (implying that linear interpolation between the end points is satisfactory), no transmission occurs. If the average distance is greater than the threshold, implying that the current frame is not part of the previous trend, the previous frame is transmitted. Thus, the new algorithm finds suitable end points for line segments that reasonably approximate the computed LARs. These end points are then transmitted.

We evaluated the speech quality produced by the real-time vocoder using the new Variable Frame Rate (VFR) (optimal-linear-fit) algorithm. We found, using RTFUD, that the vocoder using the older form of VFR transmission transmits continuous speech at a rate of about 2500 bits/second, using the nominal value for the VFR threshold. At a transfer rate this high, we hear little, if any, quality improvement with the new VFR algorithm. However, for transmission rates below 2000 bits/second, the new method produces speech that is clearly superior to that produced by the old method, as shown in Figure 2.6. The quality produced by the old method degrades rapidly as the transmission rate is reduced beyond a certain point (approximately 2000 bits/second for continuous speech). The new method degrades more slowly, and continues to degrade slowly even when the transmission rate is reduced beyond this point. Since the transmission rate and the quality of the speech depend on the specific utterance, the bit rates discussed are approximate and should be understood to be only

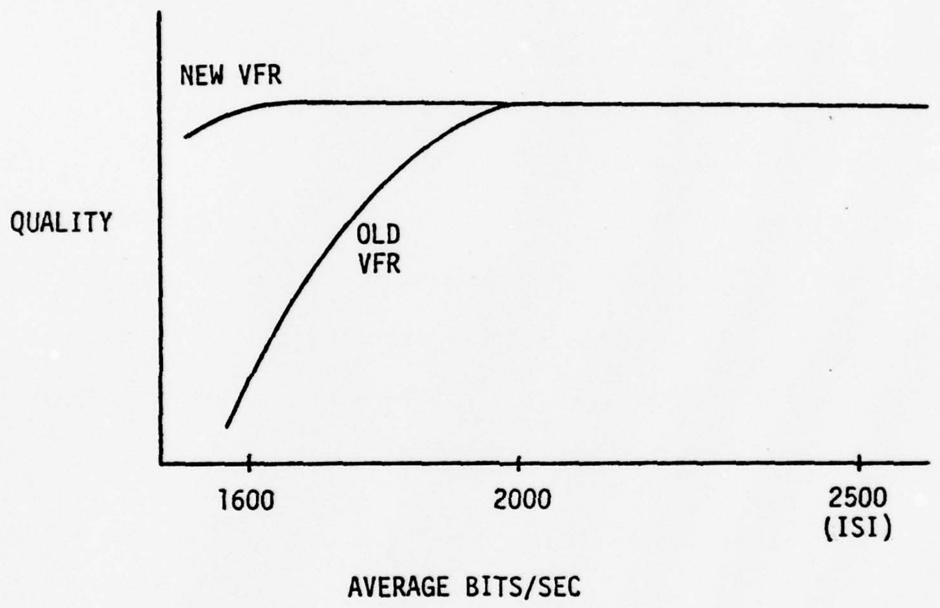


Fig. 2.6 Speech Quality Comparison of Old VFR and New VFR Algorithms

indicative of the general trend shown in Figure 2.6.

2.2.4 Mixed Source Analysis and Synthesis

During the past year we have added mixed-source analysis and synthesis to the real-time vocoder.

2.2.4.1 Motivation

The commonly used source or excitation for the synthesizer of narrowband LPC vocoders is the result of an idealized model, which is either a sequence of pulses separated by the pitch period for voiced sounds, or white noise for fricated (or unvoiced) sounds. The major deficiencies of this model are two-fold: (1) Some speech sounds, e.g., voiced fricatives such as [z], are produced using both vocal cord vibrations and turbulent noise as excitation for the vocal tract; (2) Errors in the binary voiced/unvoiced (V/UV) decision are readily perceived by listeners as a severe degradation to the quality of the synthesized speech.

The new source model that we have added to the real-time vocoder has both the voiced (pulse) source and noise source; it allows for selective excitation of different speech frequency bands by different sources. The spectrum is divided into a low frequency and a high frequency region, with the pulse source exciting the low region and the noise source exciting the high region. The cut-off frequency that separates the two regions is a parameter of the model, which is computed and transmitted to the receiver. The cut-off frequency is a continuous rather than a binary parameter, and it has a "soft-fail" effect on perception in that perception is not very sensitive to small changes in its value. (For more on the mixed-source model, see the paper in Appendix B.)

2.2.4.2 Details of the Algorithm

Mixed-source analysis begins with the pitch and voicing estimates obtained from the input speech by the SIFT pitch extraction algorithm in the real-time vocoder. The input data originally used by SIFT is multiplied by a Hamming window centered on the data, whose length is determined by the pitch estimate, such that the window contains 2 to 3.5 pitch periods. The length of the window is important. If it is too narrow, the harmonics cannot be resolved; if it is too wide, local maxima unrelated to the harmonics can occur.

Once the data is suitably windowed, an FFT is performed to obtain a 256-point magnitude-squared spectrum. A smaller FFT would yield resolution too coarse for resolving the harmonics of low-pitched speech. The algorithm then finds all local maxima, using 3-point quadratic interpolation to find the frequency of each peak. The harmonics are then examined to determine the frequency at which the harmonic structure disappears. The harmonic structure may disappear over a portion of the spectrum and then reappear at a higher frequency. The algorithm allows for this possibility and chooses the cutoff frequency to be the highest frequency at which harmonic structure is evident. If the frame is unvoiced, the cutoff frequency is set to zero. The cutoff frequency is quantized to 3 bits and transmitted only when it changes.

The synthesis portion of the algorithm applies a low-pass filter, with cutoff frequency corresponding to the transmitted cutoff, to the pulse excitation, and a high-pass filter, with the same cutoff, to the noise excitation. The filtered excitations are added, and then passed through the standard synthesis all-pole filter.

2.2.4.3 Results

The mixed-source analysis and synthesis algorithm requires an additional 1Ø57 (octal) words of program source memory over the 143Ø words required by the standard LPC algorithm. It also requires 6.2 msec of execution time.

In the AP-12ØB implementation, the algorithm behaved much as expected. During sonorant speech, the cutoff value was usually 7, with some occurrences of 6, implying fully voiced. During voiced fricatives, the cutoff value dropped to between 2 and 4. However, informal listening tests showed that there was little perceptual difference between the speech generated by the standard vocoder and the vocoder incorporating the mixed-source model modifications. We believe that the expense (in terms of execution time and program source memory) incurred by the new algorithm is not justified by the perceptual improvement obtained.

Our earlier work with the mixed-source model was done in the context of a vocoder with a 5 kHz bandwidth. The real-time vocoder has a 3.3 kHz bandwidth. We believe that the mixed-source model is most effective in improving quality in the frequency region that is missing in the real-time vocoder. If the bandwidth used by the real-time vocoder is ever increased, we believe that the mixed-source model could have a positive effect on the resultant speech quality.

3. ANALYSIS AND CODING

During the past year we investigated alternative techniques for estimating and coding the spectral parameters used in the LPC vocoder. Section 3.1 presents our work in trying to improve the estimation of LPC parameters using adaptive lattice and autocorrelation methods. Section 3.2 describes our efforts at spectral coding of LPC parameters.

3.1 Adaptive LPC Analysis

In adaptive analysis, the spectral parameters are updated every sample. Sets of these parameters are then selected for transmission. Because of the parallel repetitive nature of adaptive analysis algorithms, they are well suited for implementation in hardware.

We have investigated two such adaptive algorithms; the adaptive lattice method described by Makhoul and Viswanathan in [2] and in Appendix A, and the adaptive autocorrelation method described by Barnwell [3].

3.1.1 Adaptive Lattice Analysis

Fig. 3.1 shows the basic lattice structure that is used in LPC analysis. $x(n)$ is the input speech signal, $f_m(n)$ is the "forward" residual at stage m, $g_m(n)$ is the "backward" residual, and K_m is

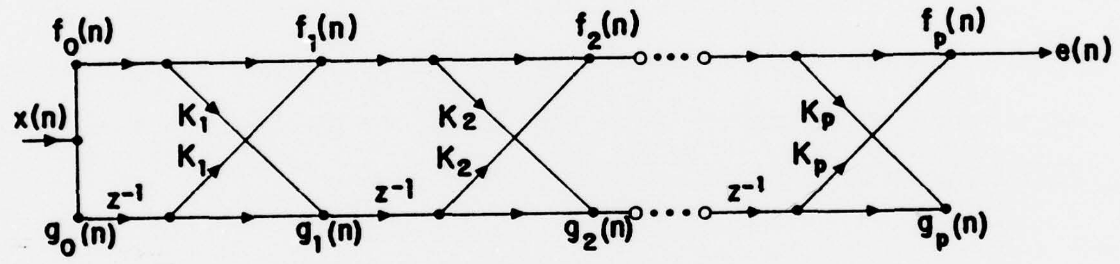


Fig. 3.1 Basic lattice structure used in LPC analysis

the reflection coefficient. Let the forward transfer function from the input to the output be

$$A_p(z) = 1 + \sum_{k=1}^{p} a_k z^{-k} \tag{1}$$

where p is the number of stages in the lattice. Then $1/A_p(z)$ is the all-pole filter used in the speech synthesis at the receiver. For the all-pole filter to be stable one can show that the reflection coefficients must obey

$$|K_m|, \ 1 \leq m \leq p. \tag{2}$$

In a situation such as speech, the signal x(n) is nonstationary, and therefore the filter coefficients K_m must vary as a function of time. From Fig. 3.1, the following time-varying relations hold

$$f_\emptyset(n) = g_\emptyset(n) = x(n) \tag{3a}$$

$$f_m(n) = f_{m-1}(n) + K_m(n) g_{m-1}(n-1) \tag{3b}$$

$$g_m(n) = K_m(n) f_{m-1}(n) + g_{m-1}(n-1) \tag{3c}$$

where we have shown the explicit dependence of K_m on time as $K_m(n)$. Assuming that we know all the quantities in (3) at time n, we need to compute the value of $K_m(n+1)$ at time n+1.

The computation of K_m is based on the minimization of a mean-square type of error of the form:

$$E_m(n) = \sum_{k=-\infty}^{n} w(n-k)\, e_m^2(k) \qquad (4)$$

where $e_m^2(k)$ is a function of the forward and backward residual energies given by

$$e_m^2(k) = (1-\gamma) f_m^2(k) + \gamma g_m^2(k), \quad 0 \leq \gamma \leq 1 \qquad (5)$$

and w(n) is a weighting sequence, or window, that weights the residual energy into the past. For an adaptive situation, one designs w(n) such that the more recent values are given greater importance. The constant γ determines the mix between the forward and backward residuals. The value of K_m is obtained by minimizing (4) with respect to K_m. The result is

$$K_m(n+1) = - \frac{\sum_{k=-\infty}^{n} w(n-k) f_{m-1}(k) g_{m-1}(k-1)}{\sum_{k=-\infty}^{n} w(n-k) [\gamma f_{m-1}^2(k) + (1-\gamma) g_{m-1}^2(k-1)]} \qquad (6a)$$

$$= - \frac{C_m(n)}{D_m(n)} \qquad (6b)$$

Theoretically $K_m(n+1)$ in (6) is guaranteed to obey (2) only for $\gamma = 0.5$. However, in practice, a wider range of γ can be used without violating (2).

Windowing

The window w(n) must have the property

$$w(n) = 0, \quad n<0 \qquad (7)$$

so that only past values of the forward and backward residuals are used in computing (6). Therefore, from (6), we see that w(n) can be considered as the impulse response of a causal filter. If the filter is *recursive and of finite order*, the numerator and denominator can be computed recursively.

In our experiments we used real-pole filters of the form

$$W(z) = \frac{1}{(1-\alpha^2 z^{-1})^N}, \quad 0<\alpha<1, \tag{8}$$

where we have used the z transform notation. These are multiple-pole filters determined by two parameters: N, the order of the filter, and α^2, the pole location. As an example, assume N=1, then the numerator and denominator in (6b) can be computed recursively from

$$C_m(n) = \alpha^2 C_m(n-1)+f_{m-1}(n)g_{m-1}(n-1) \tag{9a}$$

$$D_m(n) = \alpha^2 D_m(n-1)+[\gamma f_{m-1}^2(n)+(1-\gamma)g_{m-1}^2(n-1)] \tag{9b}$$

Procedure

The procedure for computing K_m adaptively is as follows. At time n, we assume to have in computer memory the following quantities

$$\begin{gathered} K_m(n),\ 1 \le m \le p \\ g_m(n-1),\ 1 \le m \le p \\ C_m(n-i),\ D_m(n-i);\ 1 \le m \le p,\ 1 \le i \le N \\ x(n) \end{gathered} \tag{10}$$

where p is the order of the LPC filter in (1) and N is the order of the window filter in (8). The procedure then is:

1. $m \leftarrow 1$
2. Compute the residual values $f_m(n)$ and $g_m(n)$ using (3) and (10).
3. Compute $C_m(n)$ and $D_m(n)$ recursively. (Use (9) for N=1, for example)
4. Compute $K_m(n+1)$ from (6b).
5. $m \leftarrow m+1$
6. If m>p, exist; otherwise, go to step 2.

The experimental results will be given in Section 3.1.4.

3.1.2 Adaptive Autocorrelation Analysis

In this method [3], the short-term autocorrelation coefficients of the signal x(n) are computed recursively from:

$$R_m(n) = \sum_{k=-\infty}^{n} w_m(n-k)\,x(n)\,x(n-m),\ 0 \le m \le p \tag{11}$$

where $R_m(n)$ is the mth autocorrelation coefficient at time n and $w_m(n)$ is a window that weights the lagged products x(n)x(n-m) into the past. The autocorrelations can then be used to compute the LPC coefficients at any time n using the normal equations

$$\sum_{i=1}^{p} a_i(n) R_{m-i}(n) = -R_m(n), \quad 1 \le m \le p \; . \tag{12}$$

The reflection coefficients $K_m(n)$ are obtained as a byproduct of solving (12).

We used two window filters: a one-pole filter and a three-pole filter

$$W_m(z) = \frac{\alpha^m}{1-\alpha^2 z^{-1}}, \quad 0<\alpha<1 \tag{13}$$

$$W_m(z) = \frac{\alpha^m[(1+m) + (1-m)\alpha^2 z^{-1}]}{(1-\alpha^2 z^{-1})^3}, \quad 0<\alpha<1 \tag{14}$$

Note the difference between the windows here and those used for the lattice in (8). The window for the lattice is the same for all lattice stages. Here, the window is different for different autocorrelation coefficients; that is why $W_m(z)$ is subscripted to indicate its dependence on the autocorrelation index.

3.1.3 Selection of Parameters for Transmission

Because a new set of parameters is computed for each input sample, some method must be employed to select the sets to be transmitted in order to minimize the data rate. We have investigated three such methods.

The first and simplest method is to send every Mth set, where M corresponds to the number of samples in the interframe transmission interval. This is equivalent to "sampling" the parameters once each frame.

The second method involves low-pass filtering and downsampling the discrete-time signals composed of the values of individual parameters. Since this method showed no improvement over the sampling method and was much more expensive computationally, we postponed more detailed investigation of it.

The third method attempted to find the "best" set of parameters in a frame. Since the values of the parameters vary with the pitch pulse, it is desirable to use the set of parameters corresponding to a constant time interval relative to the pitch pulse. We attempted to find these sets of parameters by identifying the instant of time during the frame when V_p, the normalized error, was minimum. The simple minimum proved to be inadequate, however. The resultant speech was judged to be "wobbly" indicating rapid variations in the parameters from frame to frame.

3.1.4 Experimental Results

In both the lattice and autocorrelation methods, we "transmitted" nine reflection coefficients once every 9.6 ms, with no quantization.

There is one parameter, γ, in the lattice method that does not exist in the autocorrelation method. We tried three values of γ: 0, 0.5, 1. $\gamma=0$ corresponds to minimizing the forward residual energy; $\gamma=1$ corresponds to minimizing the backward residual energy; and $\gamma=0.5$ corresponds to minimizing the average of the two. In our experiments we found that $\gamma=1$ gave the best quality, though the differences between the three cases were small. The results below apply to the case where $\gamma=1$.

The only variables that remain are the window parameters: α and the window order. Two window orders were used: 1-pole and 3-pole windows (N=1 and 3 in (8) for the lattice method, and the windows (13) and (14) for the autocorrelation method). For each of the two window orders the value of α was varied to optimize the speech quality.

As the value of α was increased, the speech quality changed from being crisp but rough and wobbly, to smooth but reverberant and muffled. For each case, there was a value of α for which these effects were minimal and the speech quality was judged to be the highest. The adaptive lattice and adaptive autocorrelation methods gave the same speech quality for a certain window order and a certain α. Fig. 3.2 shows the optimal ranges of α for which the speech quality was judged to be the best. Also shown are the preferred values of α. For the 1-pole cases, the optimal range for

Window	Optimal α Range	Preferred α Value
1-pole	0.992-0.994	0.993
3-pole	0.975-0.980	0.978

Fig. 3.2 Optimal ranges for α for the 1-pole and the 3-pole windows for both the lattice and autocorrelation methods.

		3-pole		
		α = .972	.975	.978
1-pole	α=.992	19	23	24
	.993	20	22	23
	.994	22	22	24

6 sentences, 4 listeners
maximum score in each bin = 6 x 4 = 24

Fig. 3.3 Number of times 3-pole method preferred over 1-pole method for three preferred values of α for each method.

α was 0.992-0.994, with 0.993 being preferred. For the 3-pole cases, the optimal range for α was 0.975-0.980, with 0.978 being preferred.

In the optimal ranges shown in Fig. 3.2, the 3-pole windows almost always produced higher speech quality than 1-pole windows. Fig. 3.3 shows the number of times the 3-pole window was preferred over the 1-pole window, using the lattice method with 6 sentences from male and female speakers. The scores were obtained from four listeners. Therefore, we recommend the use of 3-pole windows over 1-pole windows.

When compared to the standard, finite-window LPC, the adaptive methods with 3-pole windows gave slightly better quality, especially during sonorant regions.

3.2 Spectral Coding of LPC Parameters

It is well known that channel vocoders possess frequency specificity, in that the channels are independent and the quantization of one channel does not affect other channels. This property is especially important in the presence of acoustic noise. During the past year, we have attempted to develop a type of coding for LPC parameters that we hoped would have similar properties.

We argued in our previous work that the reflection coefficients and the poles are the only two sets of parameters (from a large set that we investigated) that guarantee filter stability under quantization. We chose the reflection coefficients for transmission because they are naturally ordered, and the poles are not. (Natural ordering is very important in taking statistics and using them to reduce the transmission rate by proper encoding.)

However, the poles continue to have one advantage that is not shared by other parameters, namely that the poles possess frequency specificity. This is important for two reasons: 1) The ear is especially sensitive to formant positions, which have largely one-to-one correspondence with pole positions, and 2) The sensitivity of the ear is frequency selective, namely, it is more sensitive to low frequencies than to high frequencies. Therefore, had it not been for the natural ordering problem, the poles would have been prime candidates for transmission.

It is important to note that when any reflection coefficient is quantized, the result is a perturbation of the spectrum at all frequencies, with no control over any particular region. On the other hand, if a pole is quantized, the effect is largely seen in the region of the formant corresponding to that pole; the positions of other formants are not affected at all. We believe that this frequency specificity is important in maintaining high speech quality.

We implemented a new spectral coding scheme which transforms the set of p poles inside the unit circle to a set of p poles on the unit circle. In this manner, the new poles become automatically ordered by frequency (i.e., position on the unit circle), and hence one can collect the statistics needed for efficient coding.

The new poles are actually computed from two polynomials that are easily obtained form the original LPC polynomial [4]. Let the LPC polynomial of order p be given by (1). The LPC polynomial of order p+1 is given by

$$A_{p+1}(z) = A_p(z)+K_{p+1}z^{-(p+1)}A_p(z^{-1}) \qquad (15)$$

where K_{p+1} is to be specified. The method consists of setting K_{p+1} to either +1 or -1, resulting in two polynomials:

$$A^{+}_{p+1}(z) = A_p(z)+z^{-(p+1)}A_p(z^{-1}) \qquad (16)$$

$$A^{-}_{p+1}(z) = A_p(z)-z^{-(p+1)}A_p(z^{-1}). \qquad (17)$$

The coefficients of $A^{+}(z)$ and $A^{-}(z)$ are thus obtained from the coefficients of $A_p(z)$ by simple additions and subtractions. The two new polynomials are guaranteed to have their poles interleaved on the unit circle. (The actual implementation involves the root computation for two transformed polynomials, each having p/2 roots on the real line between -1 and 1, which is much simpler than computing the p+1 complex roots of each of the two polynomials.)

Figure 3.4 shows an example where the crosses on the unit circle represent the roots of $A^+(z)$, the circles represent the roots of $A^-(z)$, and the roots of $A_p(z)$ are inside the unit circle. (The crosses and circles on the real line are the projections of those on the unit circle.)

At the synthesis end, $A_p(z)$ is computed from (3) and (4) by simple addition:

$$A_p(z) = [A^+_{p+1}(z)+A^-_{p+1}(z)]/2. \tag{18}$$

Unfortunately, we found that the spectral coding technique did not have the desired frequency specificity property. A quantization error in any of the poles on the unit circle affected the locations of <u>all</u> of the transformed poles inside the unit circle. The general result was that this spectral coding method did not result in substantially improved quality over current coding techniques.

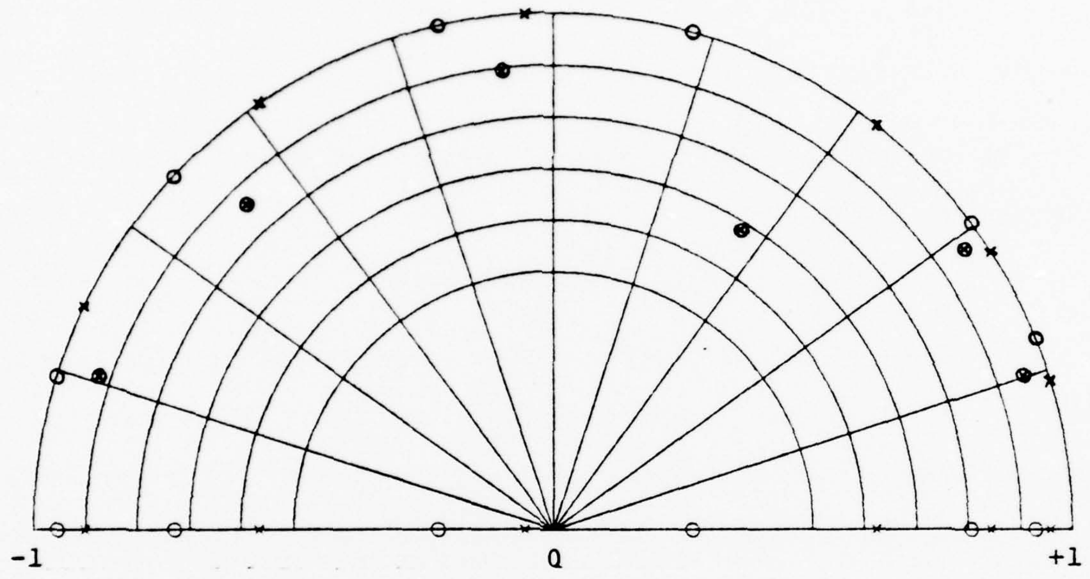


Fig. 3.4 Pole distribution in the z-plane for a single LPC all-pole spectrum with p=12. The actual poles are <u>inside</u> the unit circle. The result of the proposed pole transformation is shown <u>on</u> the unit circle, with the crosses showing the roots of $A^+(z)$ and the circles the roots of $A^-(z)$. The projections of those roots on the real line are also shown.

4. SOURCE MODEL FOR RESIDUAL-EXCITED CODERS

In this section we report on our work on the residual-excited baseband coder. Most of the effort has gone into developing a source model for use as excitation to the synthesis filter of a baseband coder. In the following subsections we describe the baseband coder and point out the need for source modeling. We then summarize our research in that area.

4.1 A General Residual-Excited Baseband Coder

In a baseband coder, a low-frequency portion of the signal, known as a baseband, is quantized and transmitted. Figure 4.1 shows the block diagram of a residual-excited baseband coder. The system is based on a linear prediction representation of the speech signal. In Fig. 4.1, the block labelled "baseband extraction" has two functions. The first is to filter the input speech signal by means of the LPC inverse filter. The second function is to lowpass filter and downsample the residual waveform to retain its low frequency components. As shown in the figure, a major task at the receiver is to regenerate the missing high-frequency components. The output of the high-frequency regeneration box is an excitation signal with a flat spectrum. In the absence of quantization noise, the low frequency portion of the excitation signal is identical to the baseband residual while the high-frequency portion is the

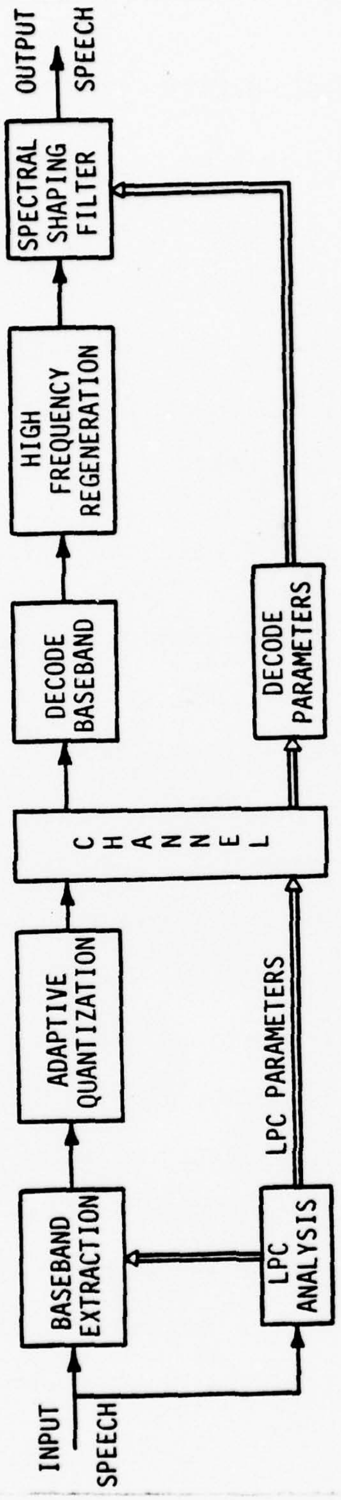


Fig. 4.1 General residual-excited baseband coder

result of modeling the original residual spectrum. The fullband excitation signal is used as input to the all-pole LPC synthesis filter to generate the output speech.

The quality of the output speech signal is determined by four factors: a) width of the baseband, b) coding of the baseband, c) estimation and coding of spectral parameters, and d) the high-frequency regeneration (HFR) method employed. In our work to date we have concentrated mainly on the fourth factor, HFR. The HFR method will be explained below. As for the coding of the baseband--an important consideration for the design of the coder--we have thus far chosen a simple approach, that of APCM of the baseband residual together with entropy coding to maintain the total average bit rate at or below 9.6 kb/s.

4.2 High-Frequency Regeneration

In the second quarterly progress report (QPR) and in a recently published paper, *included here as Appendix C, we have* presented new methods of high-frequency regeneration for baseband coders. The idea behind the new methods derives from the pitch-excited LPC vocoder. In voiced excitation, the spectrum of the excitation is a flat line spectrum at multiples of the fundamental pitch frequency. Such a spectral structure is periodic and repetitive: the high-frequency structure is the same as at low frequencies. The spectrum of unvoiced excitation, on the other hand, is continuous and has a random spectrum with a flat envelope. However, the details of the unvoiced spectrum are not as perceptually important as the details of the voiced spectrum. Therefore, the unvoiced spectrum can be considered repetitive also, in that any similar spectrum can be substituted with equally good results.

The new regeneration method, then, is simply to duplicate the baseband spectrum at higher frequencies, in some fashion. We discussed two types of HFR: spectral folding and spectral translation. Figure 4.2 illustrates the effect of the two types of HFR for the 3-band case, with a baseband width of B Hz. In spectral translation, the frequency components between nB and (n+1)B are simply a highpass translated version of the frequency components between Ø and B, for all n. Such is also the case for spectral folding but only for n even. In spectral folding and for n odd, the frequency region between nB and (n+1)B is the mirror image of the baseband.

4.2.1 Time-Domain HFR

We now describe the time-domain implementation of these methods. Briefly, integer-band spectral duplication is the case where the baseband width, B Hz, is adjusted such that the total

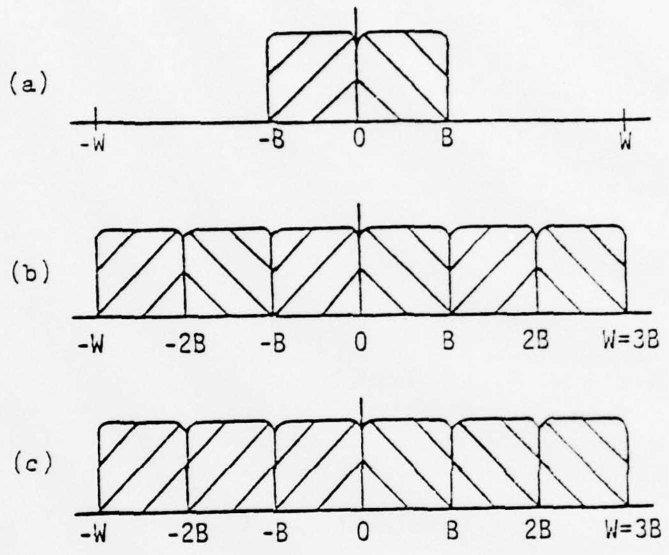


Fig. (a) Baseband spectrum
(b) 3-band spectral folding
(c) 3-band spectral translation.

Fig. 4.2 Illustration of High-Frequency Regeneration by Spectral Duplication

signal bandwidth, W Hz, is an integer multiple of B, i.e., W=LB, with L an integer (See Fig. 4.2). The time domain implementation of integer-band HFR by spectral folding is simply done by inserting L-1 zeros between consecutive time-domain samples of the baseband residual. Spectral translation is implemented in a similar fashion, but it requires further filtering. For details on both methods, the reader is referred to Appendix C and to the second QPR. Spectral folding and translation are attractive because, in general, they are computationally less expensive than the traditional waveform rectification methods. Waveform rectification has been the most commonly used HFR method in baseband coders.

In preliminary experiments using HFR by spectral folding or spectral translation we heard a number of distortions in the form of added tones. These tones were generally more audible for a smaller baseband width and for higher-pitched voices. For the spectral folding case, we were able to verify the existence of a tone at even multiples of the folding frequency, i.e. at multiples of 2B Hz. The tone was largely eliminated by a simple method: we subtracted off the short-term d.c. in the baseband residual, because the d.c. is folded into multiples of 2B Hz. Following spectral folding, we restored the original d.c. so as not to disturb the average signal level, but the energy at multiples of 2B Hz had already been eliminated.

Other audible tones have been more difficult to trace. However, HFR by integer-band spectral duplication does not seem to cause any perceptible roughness, as was the case in rectification.

One reason for the existence of these background tones may be the fact that, with spectral duplication, the harmonic structure is interrupted at multiples of B Hz. We hypothesized that the tones may be eliminated by adjusting the width B of the baseband to be a multiple of the pitch fundamental frequency on a short-term basis. Unfortunately, if implemented in the time domain, such a scheme would require an enormous amount of computation which would offset the initial reduction in computation afforded by the spectral duplication method. We have therefore decided to take a frequency domain approach that would allow us to implement pitch-adaptive HFR with a minimal amount of computations. This is explained next.

4.2.2 Frequency-Domain HFR

In the frequency domain, the baseband frequency components can be easily duplicated at higher frequencies to obtain the frequency components of the full-band excitation signal. As mentioned earlier, in our work to date, we have transmitted the baseband residual waveform using APCM. Thus, at the receiver, it is necessary to perform a time-to-frequency transformation prior to HFR. Once in the frequency domain, care is taken not to interrupt the harmonic structure of the excitation signal by using an estimate of the pitch. In case the coding at the transmitter makes use of pitch, the value of pitch is transmitted and is readily available at the receiver. Otherwise, the receiver can easily extract a pitch value from the received baseband, e.g., by detecting the location of the peak of the autocorrelation of the baseband. With pitch known, the receiver extracts a subinterval of the baseband containing an integer number of harmonics. The chosen subinterval is duplicated (translated) at higher frequencies as many times as necessary to fill the missing frequency components. The HFR procedure is illustrated in Fig. 4.3.

For voiced sounds, we found that good perceptual results are obtained when the subinterval extends from the spectral valley just before the first harmonic to the valley just after the last complete harmonic present in the baseband (see Fig. 4.3). The upper frequency edge of the subinterval is C Hz and is pitch-dependent. For unvoiced sounds, the subinterval consists of the whole baseband, less its two end points: the d.c. component and the component at B Hz. Following the HFR process, a frequency-to-time transformation yields the full-band time-domain excitation signal to be applied to the synthesis filter $1/A(z)$. We note here that the effective baseband width is $C<B$. The received frequency components between C and B are discarded, and those between C and W are regenerated (See Fig. 4.3).

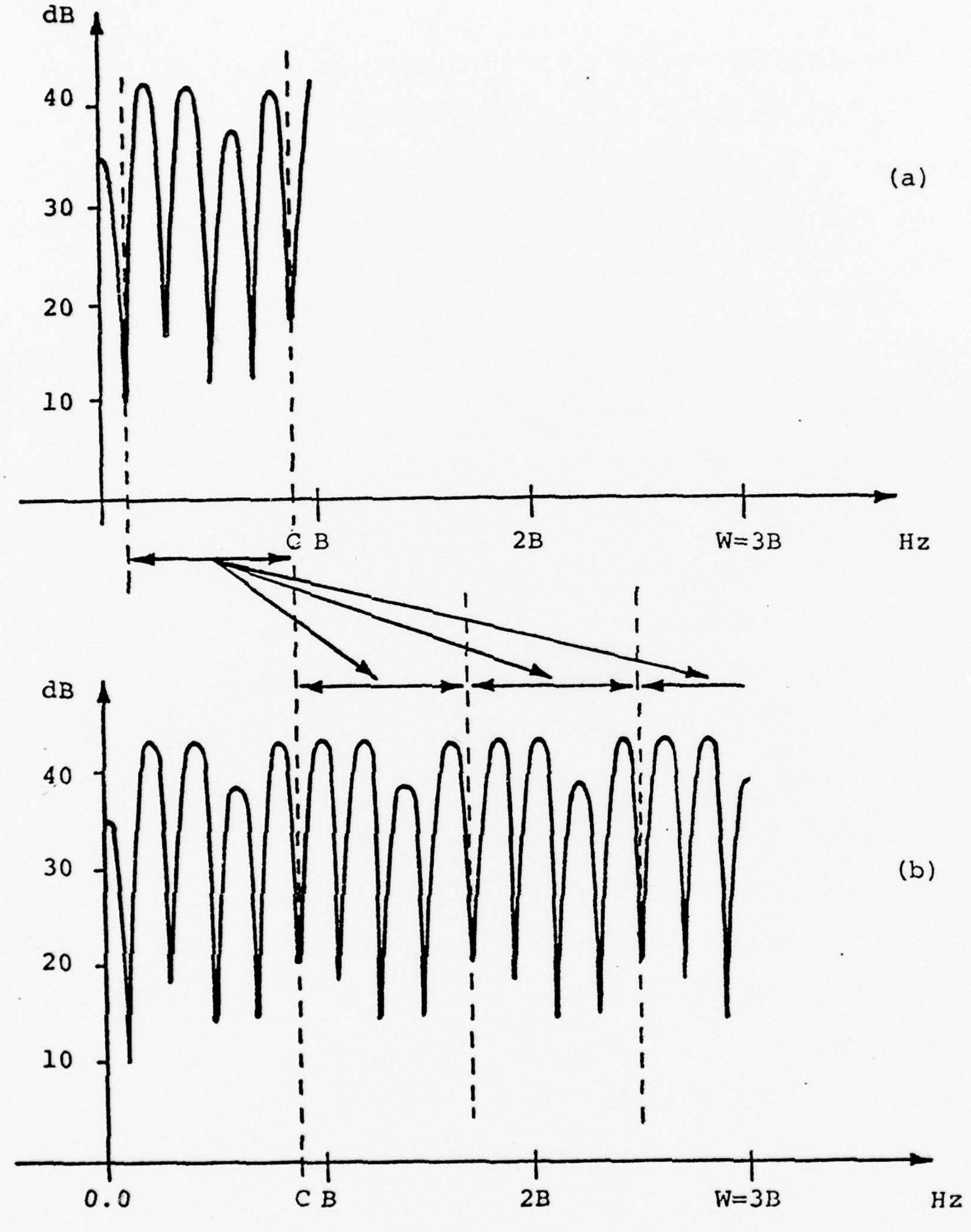


Fig. 4.3 Pitch-Adaptive Frequency-Domain High-Frequency Regeneration

(a) Received baseband spectrum
(b) Full-band regenerated excitation spectrum

One possible extension of the above described HFR method is to let the transmitter locate and extract the subinterval of the baseband to be used in the HFR process. In such a case the transmitted baseband would be pitch-adaptive and of width C Hz.

A second, and perceptually more important extension of the method, provides for a better placement of the frequency-translated intervals. In the above described method of HFR, we assumed that the spectrum of voiced sounds is periodic in frequency. However, we know that speech spectra are not exactly harmonic in structure. To take into account the irregularities of the speech spectrum, we shift the high-frequency interval around its nominal position in such a manner as to match, as best as possible, the corresponding original frequency components of the full-band residual. This task is done at the transmitter, where the full-band excitation spectrum is available. First, the chosen subinterval is translated to its nominal high-frequency position. Then, it is cross-correlated with the corresponding frequency components of the full-band residual. The "optimal" location is then chosen to be at the positive maximum value of the cross-correlation. When short correlation lags are considered, i.e., between 0 and 3, the additional cost is only 2 bits for each frequency-translated interval. The transmitted information indicates to the receiver where to place the baseband subinterval in the high frequency region, relative to its nominal position.

4.3 Results

We have implemented a preliminary version of the proposed baseband coder and simulated its operation at a transmission rate of 9.6 kb/s. For inputs at a sampling rate of 6.67 kHz, we have chosen the frame size to be 19.2 ms, i.e., 128 samples. In our experiments thus far, we transmitted the baseband residual using time-domain APCM with entropy coding. The system parameters (LPC and gain) are transmitted separately at the rate of 2.3 kb/s, leaving about 7.3 kb/s for the baseband residual.

In our initial experiments, we implemented frequency-domain integer-band spectral folding and spectral translation. At the receiver, we tried both the discrete Fourier transform (DFT) and the discrete cosine transform (DCT). We found the frequency-domain results to be perceptually similar to the time-domain results, with low-level tones and no roughness in the background. In more recent experiments, we performed pitch-adaptive HFR with the added cross-correlation feature for better spectral duplication (using only the DCT). Upon informal listening, we found that this system provides a marked improvement in speech quality over the traditional waveform rectification approach and over the non-pitch-adaptive time-domain spectral duplication methods. However, the method introduces some roughness reminiscent of waveform rectification.

We expect the quality of the synthetic speech to improve further when we change the coding of the baseband from APCM to adaptive transform coding (ATC). The idea here is to perform the requisite time-to-frequency transformation at the transmitter prior to coding. Our expectations of improvement in quality are based on the fact that ATC provides an increase in SNR over APCM and lends itself more readily to spectral noise shaping which is essential to minimize the perception of quantization noise. We have chosen the DCT because it fits in with existing ATC schemes. In fact, we began the implementation of ATC of the baseband residual and we shall discuss it in detail in the next quarterly progress report since work is still on-going at this point.

In our future work, we shall look into the possibility of having a multirate coding system. The system bit rate can be varied by varying the width of the baseband. Alternatively, the transmitter may be operating at a fixed rate, but the transmission channel will be allowed to discard some of the bits. The bits (codes) are usually arranged to correspond to the transmitted frequency components, in an ascending order, going from low to high frequencies. Thus, the discarded codes correspond to high-frequency components. The receiver will regenerate the missing high-frequency components, using the HFR methods described above, irrespective of the actual channel rate.

5. PHONETIC SYNTHESIS

5.1 Introduction

Our phonetic synthesis development this year is an initial step in the development of a very low rate (approx. 100 bits/second) speech transmission system [5]. An overview block diagram of the very low rate (VLR) transmission system is shown in Figure 5.1. In order to achieve such a low bit rate, the VLR vocoder models the speech in terms of phoneme-sized units. The analyzer in such a vocoder would extract from a spoken sentence a sequence of triplets. Each triplet consists of a phoneme, a phoneme duration, and a single pitch value. In our synthesis work, this sequence of triplets is determined from a "target" sentence by a human transcriber. The output of the phonetic synthesis program is the complete set of "synthesis parameters" required by an LPC synthesizer. They are, for each 10 ms, 14 LPC parameters specifying a spectrum, a value of gain, a voicing flag, and (if voiced) a value of pitch and cutoff frequency for the mixed-source model [6,7]. Our goal this year has been to synthesize very natural sounding speech using only the VLR phonetic input. In addition to being part of a phonetic vocoder, this phonetic synthesis program will also be useful in a text-to-speech system, or as part of a speech storage and playback system requiring very little storage.

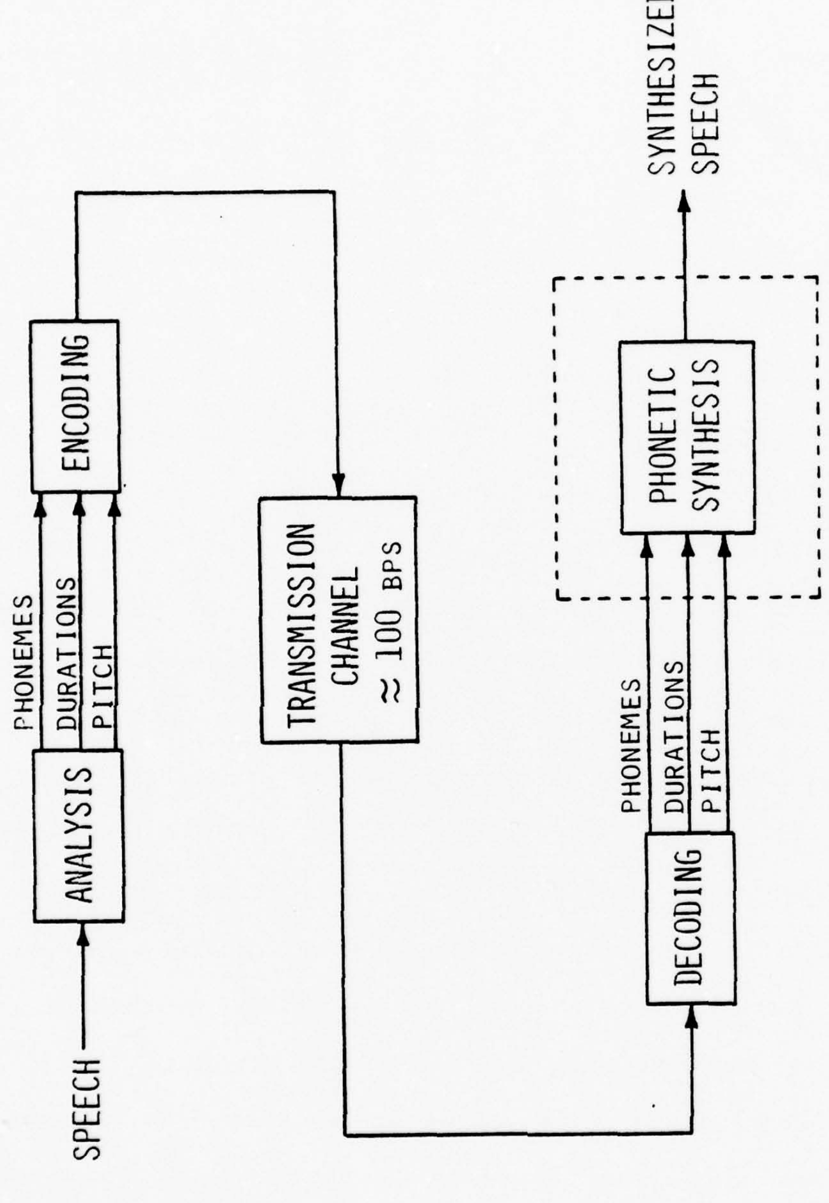


Fig. 5.1 Very low rate (VLR) phonetic vocoder

5.2 Design choices

5.2.1 Diphone templates

The basic method for phonetic synthesis that we have chosen is concatenation of diphone templates. A diphone is defined as the region from the middle of one phoneme to the middle of the next phoneme. Thus, for each possible pair of phonemes there is one diphone. A diphone template consists of the parameters necessary to synthesize that diphone.

The diphone is a natural unit for synthesis because the coarticulatory influence of one phoneme does not usually extend much further than half way into the next phoneme. Since diphone junctures are usually at articulatory steady states, minimal smoothing is required between adjacent diphones. Also, since the regions around the phoneme transitions are preserved intact, the difficult task of duplicating these transitions by complicated acoustic-phonetic rules is avoided. We estimate that approximately 2500 diphone templates are needed to achieve high quality.

5.2.2 LPC synthesis

We have chosen to use LPC synthesis because of our extensive experience with LPC analysis/synthesis systems. Consequently the diphone templates contain parameters necessary for an LPC vocoder. For every 10 ms frame in a diphone template, we store a set of 14

Log Area Ratio (LAR) parameters and a value of energy. (LAR parameters are stored instead of LPC parameters due to their better behavior under quantization and interpolation.)

5.2.3 Real Speech

A third design choice in this project has been to use diphone templates that have been extracted from real speech. It is felt that this will help in assuring that the pronunciations that result will likewise be natural. The data base from which diphone templates are extracted is described in detail in Section 5.5.

5.3 Overview

A major portion of this project consists of gathering the set of diphone templates to be used in synthesis. The diphone templates are being extracted from a carefully designed data base of short utterances spoken by a single speaker in a quiet environment. After each utterance has been digitized, the templates are specified by a researcher who indicates the appropriate phoneme boundary time markers associated with each short utterance.

Figure 5.2 illustrates the synthesis procedure. The speech synthesis program

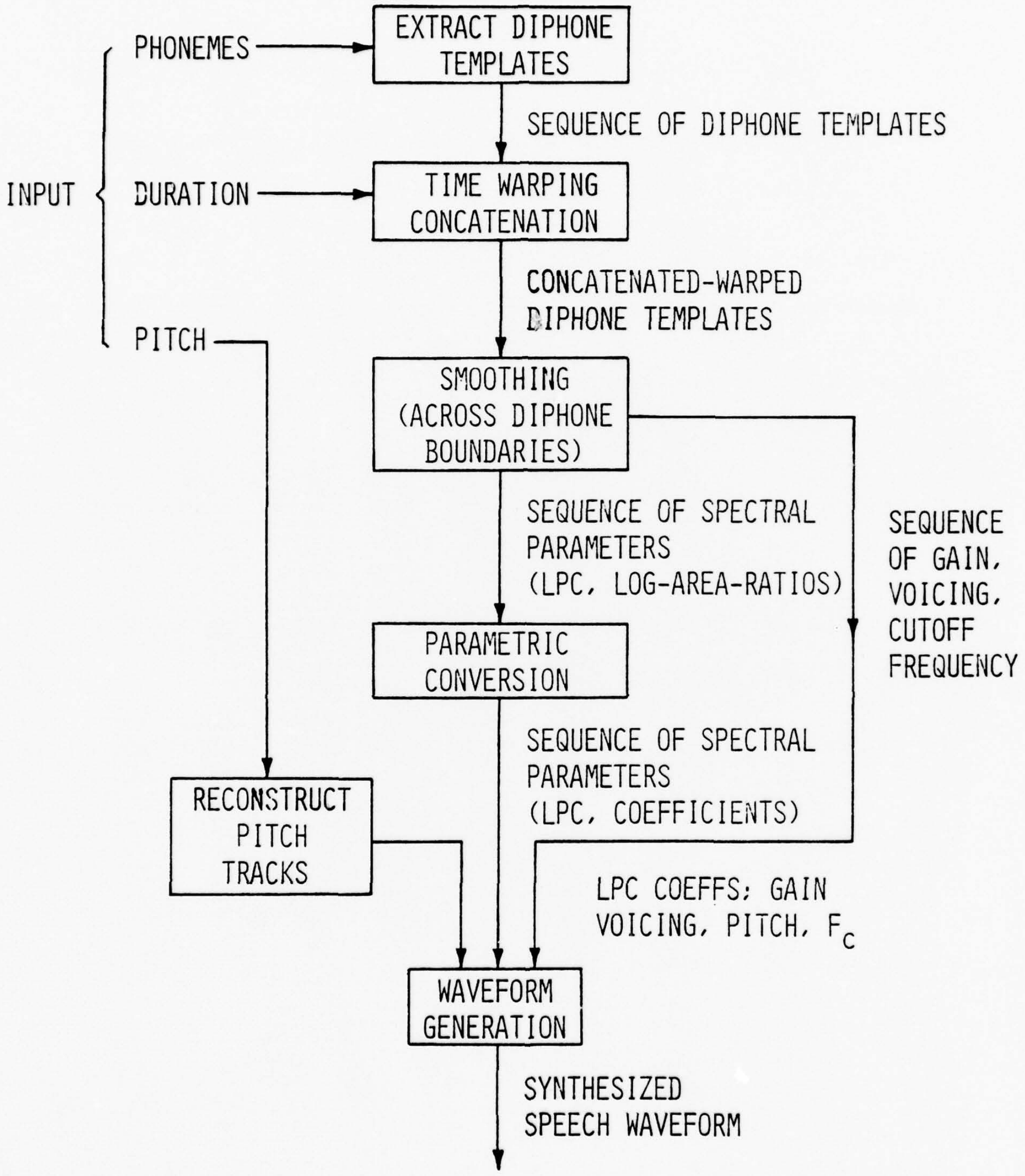

PHONEMES
EXTRACT DIPHONE TEMPLATES
SEQUENCE OF DIPHONE TEMPLATES
INPUT
DURATION
TIME WARPING CONCATENATION
CONCATENATED-WARPED DIPHONE TEMPLATES
PITCH
SMOOTHING (ACROSS DIPHONE BOUNDARIES)
SEQUENCE OF SPECTRAL PARAMETERS (LPC, LOG-AREA-RATIOS)
SEQUENCE OF GAIN, VOICING, CUTOFF FREQUENCY
PARAMETRIC CONVERSION
SEQUENCE OF SPECTRAL PARAMETERS (LPC, COEFFICIENTS)
RECONSTRUCT PITCH TRACKS
LPC COEFFS; GAIN VOICING, PITCH, F_C
WAVEFORM GENERATION
SYNTHESIZED SPEECH WAVEFORM

Fig. 5.2 Synthesis

1) translates the input phoneme sequence into a diphone sequence;

2) selects the most appropriate diphone template (depending on the local phonetic context);

3) time-warps each of the diphone templates to produce a gain track and 14 LAR parameter tracks of the specified durations;

4) smooths between adjacent warped diphone templates to minimize gain and spectral discontinuities;

5) reconstructs continuous pitch tracks by linear interpolation of the single pitch values given;

6) determines the cutoff frequency and voicing using knowledge of the phoneme being synthesized;

7) converts resulting LAR parameter tracks to LPC parameter tracks;

8) uses the resulting sequence of LPC, pitch, gain, and cutoff frequency (specified every 10 ms) as input to control an LPC speech synthesizer.

5.4 Algorithm Details

5.4.1 Extensions to Diphone Definition

In this section, we describe some extensions to the fundamental definition of a diphone which permit the highest possible intelligibility and naturalness to be attained. These extensions allow "special cases" to be handled uniformly by the synthesis program.

5.4.1.1 Context-Specific Diphones

Most diphones can be used adequately independent of context, but there are important exceptions. For instance, in the sequence [W-IH-L], as in the word "will", the part of the phoneme [IH] contained in the diphone [W-IH] is drastically affected by the presence of the [L]. Consequently, we store a separate diphone template for [W-IH] to be used in this context. To account for such phenomena we have allowed more than one template to be defined for diphones when they are affected by context. Additional diphone templates necessitated by lateralization, retroflexion and other strong contextual phenomena are expected to account for 10-20% of the total diphone inventory. The context-dependent diphone templates are determinable from a normal phoneme string.

5.4.1.2 Splitting Phonemes

Some phonemes that have more than one acoustically distinct region have been split up into two "pseudophonemes". This permits us to control the durations of each of the regions independently. For instance, the diphthong [AY], as in "bite", starts out much like an [AA], as in "pot", but ends up more like the [IY] in "beet". The two relatively steady regions are connected by a rapid transition between them. Since durations of the two steady regions are somewhat independent, as are the contextual effects of neighboring phonemes, this diphthong has been split into two pseudophonemes, [AY1-AY2], which appear only in sequence. The unvoiced plosives and affricates also have two acoustically distinct regions. Each region is treated as if it were a separate phoneme.

5.4.2 Time Warping

In order to provide input to the LPC synthesis program we must specify LPC coefficients at fixed intervals (10 ms) by time warping template information (whose duration is fixed) to satisfy phoneme durations specified by the input. This is made difficult by the fact that the time warping must preserve the naturalness of speech. One way to do this is to treat speech as being made up of elastic and inelastic regions.

The principle of distinguishing between "elastic" and "inelastic" regions of the template arises from observations of speech parameters under widely varying speaking rates. Most of the durational variation is observed to occur during the "steady state" portion of the phoneme (an elastic region), whereas the transition portions (inelastic regions) are relatively insensitive to changes in speaking rate. Hence, our time warping algorithm allows us to specify that a certain percentage of the diphone template on each side of the phoneme boundary is to be treated as relatively inelastic and the rest of the diphone template (the section corresponding to phoneme middles) as more elastic.

Time warping is accomplished by the use of piecewise linear mapping functions, which define the part of the template to be used at each instant of time. This correspondence and the resulting mapping function are illustrated in Figure 5.3. The speech being synthesized is the sequence /- DH AX M/ as in "The man...". The vertical axis represents time in the diphone templates. The horizontal axis represents time in the synthesized speech. The diphone template durations are determined from the original recorded short utterances, while the phoneme durations are determined by the input utterance to be vocoded.

The phoneme boundaries in the diphone templates are mapped onto the phoneme boundaries in the synthesized speech to define

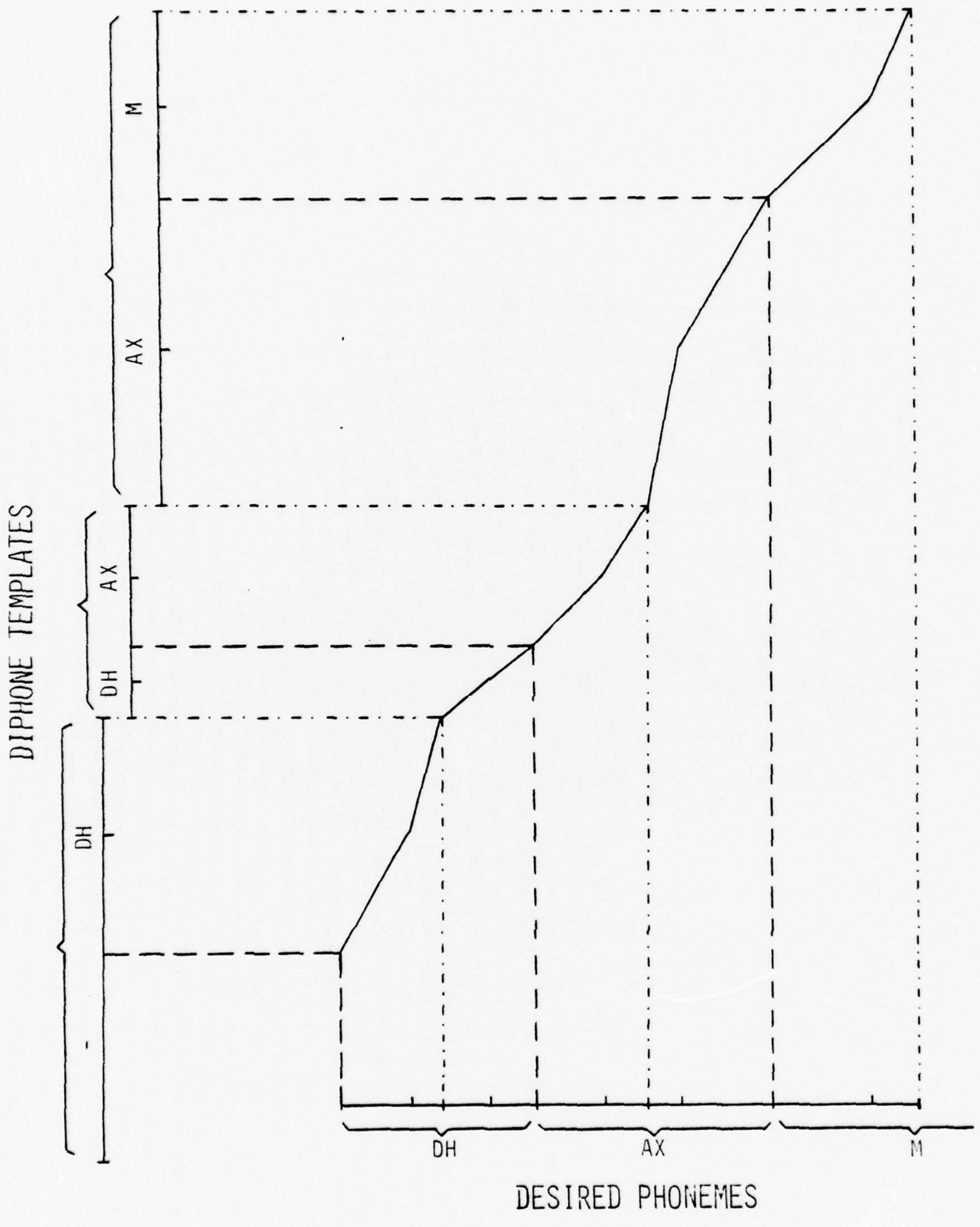


Fig. 5.3 Parameter time warping

uniquely a point in the piecewise linear mapping function. The diphone boundary is mapped onto the center of the phonemes. The DASHED lines connect phoneme boundaries, and the DASH-DOT lines connect diphone boundaries (phoneme centers). Notice that the templates are generally compressed during the mapping. The reason for this is that (whenever possible) our diphone templates are extracted from fully articulated short utterances. We deliberately designed our data base to consist of fully articulated short utterances because we believe that compressing a long template will result in better speech quality than expanding a short one. (We were also influenced by the consideration that information is more easily ignored than generated.)

The elastic and inelastic regions are delineated by small tic marks on both axes. The knees in the mapping function shown correspond to the intersection of these tic marks. Although both elastic and inelastic regions of the template (in this example) are shortened by mapping from templates to phonemes, the inelastic regions are shortened less.

As part of our test of the warping algorithm, we tried varying the rate of the synthesized speech. When the time warping was uniform, some phoneme transitions (e.g., vowel/nasal transitions) became slurred. The time warping relations were varied such that all the transitions sounded natural over a wide range of speech rates.

5.4.3 Parameter Smoothing

When templates are concatenated, discontinuities in the gain and the LAR parameters may result despite the fact that the parameters are smooth within each template. In order to deal with this, we smooth the parameters throughout an "interpolation region" that straddles the diphone template boundary (phoneme middle) and contains potential parametric discontinuities. This interpolation region is defined by two "interpolation points" (one from each diphone).

Figure 5.4 compares two different parameter smoothing algorithms as applied to a single parameter. The heavy lines indicate the parameter tracks as taken from the two diphone templates. Taken together these tracks span one phoneme. There is a discontinuity at the diphone boundary, which is indicated by the vertical DASHED line. The two vertical DASH-DOT lines delineate the interpolation region. The straight line (a) connecting the two parameter tracks illustrates the effect of linear interpolation. As can be seen in this case the linear interpolation results in a poor fit to the original data. The other curve (b) connecting the two points is derived by adding a ramp (shown at the bottom of the figure) to each parameter track, such that the discontinuity is eliminated. This second smoothing method often preserves the shape of the original parameter tracks better than linear interpolation.

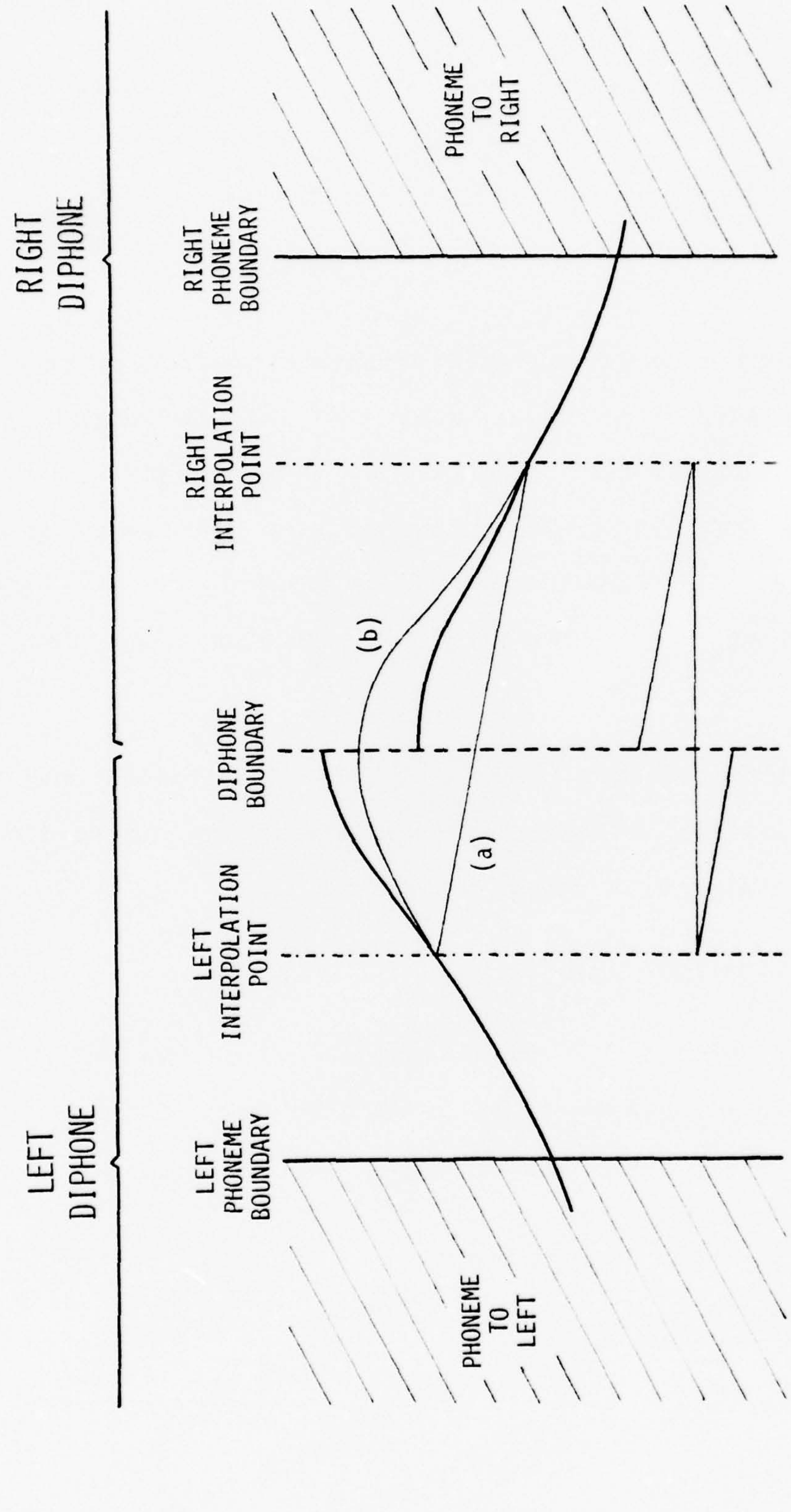


Fig. 5.4 Parameter smoothing over one phoneme

Note that the second method requires that the parameter tracks be preserved in the diphone templates. It is expected that 4 to 6 frames per diphone template will be sufficient.

5.4.4 Gain Adjustment

The energy values to be used in synthesis are stored in the diphone templates. In some cases the overall intensity of a diphone is not consistent with that of the neighboring diphones, due to differences in speaking level during the recording of the data base. This inconsistency is reduced by the optional specification of a gain adjustment for each diphone template in the data base.

If the inconsistency is due to a different intonational stress in the input speech, the adjustment to the energy during a phoneme could be specified by a stress code.

5.4.5 Excitation

In addition to 14 LPC parameters and gain, the LPC synthesis program requires for each 10 ms frame a value of pitch, a voicing flag, and a cutoff frequency.

5.4.5.1 Pitch Track

The pitch track during voiced phonemes is reconstructed from the input pitch values by straight line interpolation. In our simulation of the analysis part of a phonetic vocoder, the single transmitted pitch values are determined from complete pitch tracks in the sentence being analyzed. The analysis program (given the phoneme identities and phoneme boundaries) determines a weighted piecewise linear least-squares fit to that pitch track. The endpoints of the linear sections (which occur at phoneme boundaries) are transmitted. The weighting is designed to minimize the effect of pitch tracker errors. It was found that sentences synthesized using these piecewise linear pitch tracks are practically indistinguishable from those using the original analyzed pitch tracks.

5.4.5.2 Voicing

Voicing was determined directly from the identity of the phoneme being synthesized. Voicing errors are avoided by careful placement of phoneme boundaries in the diphone templates. We have found that a one frame error in placement of the boundary can cause a severe "pop" in the synthesized speech, due to misalignment of spectral parameters with excitation parameters.

5.4.5.3 Mixed-Source Model - Cutoff Frequency

Previous work [6,7] has shown that our mixed-source model of excitation results in more natural sounding (less buzzy) speech by allowing for specification of a cutoff frequency with every value of pitch. The voicing excitation is low-pass filtered and the frication is high-pass filtered. The cutoff frequency simultaneously marks the upper edge of the voicing spectrum and the lower edge of the frication spectrum. Currently we have implemented an algorithm that selects a cutoff frequency based on distinctive features of the phoneme being synthesized. For example, for vowels the cutoff frequency is at 5000 Hz (fully voiced); for unvoiced consonants it is 0 Hz; and for voiced fricatives (which are produced with both periodic and random excitation) it is 1500 Hz. The cutoff frequency parameter track is then low-pass filtered in time in order to minimize excitation discontinuities at phoneme boundaries. The implementation of the cutoff-frequency algorithm has resulted in a noticeable improvement in speech quality: in particular, a decrease in buzziness.

5.5 Data Base

5.5.1 Design of the Data Base

At the outset of this project we knew that the synthesis quality would depend, to a large degree, on the nature of the data base of speech from which diphone templates were to be extracted. After several experiments with different data bases, we designed a data base that seems most appropriate for this project. A significant problem with earlier data bases was that the gain parameter was not consistent between diphones that were to be abutted. This was partially due to the fact that some of the diphones were taken from the beginning of an utterance, while others were taken from the middle or end of an utterance. In addition, different diphones containing the same vowel were recorded several minutes or even hours apart, and incidental changes in speaking level and mouth-to-microphone distance produced noticeable differences in amplitude. However, there was no consistent method to determine whether one diphone was louder than another because it was inherently louder, or because the speaker just happened to be speaking louder.

Since there is evidence that gain was a major problem with the synthesized speech, the recording procedure was designed so that incidental differences in loudness could be minimized. The utterances were reorganized into groups according to the vowel phoneme of each diphone. Thus, the different diphones involving one particular vowel were all spoken within a short period (roughly 2 minutes). This close proximity helps to ensure that the speaking level was roughly constant during different instances of the same vowel.

The diphone utterances consisted of short nonsense syllables which were repeated three times as in connected speech (e.g.,[pa pa pap]). The diphones are most often extracted from the middle syllable, which tends to have a more prototypical articulation (and loudness).

In addition to short-term variations in speaking level, there is also a long-term variation in the level over several hours and between the several days that elapsed during the recordings. In order to estimate this effect, each group of utterances was initiated and terminated with a "normalization utterance". The normalization utterance we chose to use was [dædædæd]. This utterance was chosen, because the vowel [æ] in combination with a voiced plosive results in a higher amplitude than most syllables. After the data has been analyzed, the level of each diphone can be set relative to the normalization utterances, thus cancelling out long-term variations in speaking level. This framework also allows for the level adjustment that will probably be necessary for several groups of diphones.

5.5.2 Contents of the Data Base

The data base contains utterances for all the diphones that are felt to result in different acoustic patterns. The types of diphones included are shown below.

C stands for consonant; V stands for vowel

DIPHONE	EXAMPLE
CV	[pa] as in "pot"
VC	[ap] as in "top"
initial cluster	[spr] as in "spring"
final cluster	[nd] as in "and"
CC	[sf] as in "this formant"
VV	[iæ] as in "reality"

In addition to the vowels, we have included several other vowel allophones such as retroflexed vowels (vowels followed by [r]), lateralized vowels (vowels followed by [l]), [ə] (as in "about"), ɨ (as in "multiply"), syllabic nasals, and syllabic [l]. The consonants include silence, flapped [t], unreleased plosives, affricates, and glottal stops. Due to the inclusion of all permutations of these phonemes, the new recording includes a total of 1894 utterances representing 3145 diphones. However, of the 3145 diphones, a few cannot occur in English, and many more are likely not to be necessary as separate diphones. Therefore, we expect that approximately 2500 diphones will be necessary to achieve natural sounding speech.

5.5.3 Recording the Data Base

Since these recordings were to form the basis for the synthesis to be done in the remainder of this project, the recordings were monitored carefully. It was felt that, for this application, very low noise recordings were desirable. The recordings were made in a quiet room. The microphone used was an "electret" condenser microphone positioned 2 inches from the right corner of the mouth at an angle of 45 degrees to the side. The close-talking microphone was chosen over a more distant microphone because it allowed us to attenuate low level building-borne noise. Also, the quality of the microphone was judged to be quite high. The recordings were made on a Braun TG-1000 tape deck.

Each utterance (including the two normalization utterances for each group of diphones) was digitized into a single speech file using 12 bits per sample. The dynamic range of most of the utterances only requires 11 bits.

5.5.4 Labeling the Data Base

In order to extract the diphones from these relatively long (approximately 1 second) diphone utterances, we examine each utterance and indicate the identity and end points of the relevant phonemes. This transcription must be accurate since the spectra and energy parameters derived from the speech will be combined with voicing information determined from the phoneme identity. Therefore misalignment will result in apparent "voicing" errors.

Since most of the short utterances consist of three repetitions of the same syllable, the transcriber must also decide on which occurrence is most typical for each diphone. Most of the time, one of the diphones in the middle of the utterance is chosen to avoid the effects peculiar to the ends of an utterance.

A diphone extraction program converts the transcription text files into a diphone template definition text file. The program determines other necessary time points within the template by a simple set of rules. The diphone boundary is chosen as the midpoint between the phoneme boundaries. The "interpolation" points and the boundary between the elastic and inelastic regions of the diphone are defined as a percentage of the way between the phoneme boundary and the diphone boundary.

As the synthesized speech depends directly on the diphone templates used, it is essential that the different diphone templates are chosen to be compatible with each other. For example, the diphone boundaries should always be at the same articulatory position so that unnatural articulations are not introduced. One way of encouraging this is to require that all the transcribing be done by a single individual. However, since the labeling process is so tedious, we feel that it may be necessary, in the interest of time, to have a small number of people, working closely, share the labeling effort. In either case, it has become apparent, through experience, that some immediate auditory feedback is helpful.

5.6 Programs

5.6.1 Display Programs

Since a significant part of this project consists of accurately and consistently transcribing a complete set of diphones, we have modified several display programs to facilitate this effort.

Our general signal processing and display program was modified so that a user can interactively edit the manual phonemic transcriptions associated with a sentence. The program displays 10 time-varying parameters (such as energy and formants) along with user-defined manual transcriptions.

Our real-time waveform editing program was modified so that, in addition to displaying, editing and playing time waveforms, it could compute and display the instantaneous spectrum (log-magnitude power spectrum and LPC spectral envelope) corresponding to a short window of speech pointed to by a cursor on the waveform display. As the cursor is moved relative to the waveform, the program recomputes and displays the spectra (corresponding to the instantaneous cursor location) eight times per second.

The program also allows interactive manual transcription of time waveforms. The display of the speech waveform and the short-term spectra provide sufficient information for the labeling of most phoneme boundaries.

5.6.2 Compiler Programs

Several programs were written during the course of the project, which aided in managing the diphone templates extracted from the data base. Other programs were written that performed a "compiler" function to enable the synthesis program to access the proper diphone template from a large data file quickly and efficiently.

5.6.3 Synthesis Program

The synthesis program is written in a modular fashion so as to facilitate testing in several different configurations. To further facilitate testing, the input sequence of triplets can be specified by several alternative sources. For instance, the triplets can be specified directly in a text file or as a sequence of phonemes and durations typed in (with pitch values computed by rule), or indirectly by automatically generating an ordered sequence of CVC and VCV syllables, given a particular vowel.

As an aid in evaluating the performance of the individual synthesis algorithms, the synthesis program provides for specifying the source of any of the (5 types of) synthesis parameters as natural (directly from the target sentence) or synthetic (from the synthesis algorithms). Of course, if all the synthesis parameters are taken directly from the target sentence, the program becomes an LPC vocoder.

5.7 Experiments

During the course of the project there were several parts of the synthesis program that needed to be tested. When appropriate, an experiment was designed to test that part of the program. These experiments were described in the quarterly progress reports.

5.8 Project Status

Most of the algorithm development for the phonetic synthesis program is complete. Some of the time-warping and gain adjustment rules may undergo slight tuning, but no significant changes are anticipated.

The major time-consuming part of the project is involved with accumulating the large set of diphone templates. At present, the new data base has been completely recorded, and is mostly digitized. The labeling or manual transcription of the diphone templates is half completed.

We found, after transcribing half of the diphones, that some sort of auditory feedback to the transcriber was necessary, in order to assure that the time boundaries were placed correctly. We therefore augmented the diphone template compiler and the phonetic synthesis program to facilitate testing of newly transcribed diphone templates. We expect that these additions will ultimately improve the quality of the synthesized speech, and speed up the labeling process.

6. REFERENCES

[1] R. Viswanathan, J. Makhoul, and A.W.F. Huggins, "Speech Compression and Evaluation," Report No. 3794, Bolt Beranek and Newman Inc., Cambridge, Mass., April 1978.

[2] J. Makhoul, and R. Viswanathan, "Adaptive Lattice Methods for Linear Prediction," IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Tulsa, OK, April 1978.

[3] T. Barnwell, "Recursive Autocorrelation Computation for LPC Analysis," IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Hartford, CT, May 1977.

[4] F. Itakura, "Line Spectrum Representation of Linear Predictor Coefficients of Speech Signals," J. Acoust. Soc. Am., Vol. 57, Supplement No. 1, 535, Spring 1975.

[5] J. Makhoul, C. Cook, R. Schwartz, and D. Klatt, "A Feasibility Study of Very Low Rate Speech Compression Systems," Report No. 3508, Bolt Beranek and Newman Inc., Cambridge, Mass., Feb. 1977.

[6] J. Makhoul, R. Viswanathan, R. Schwartz, and A.W.F. Huggins, "A Mixed-Source Model for Speech Compression and Synthesis," 1978 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Tulsa, Okla., April 10-12, 1978, pp. 163-166.

[7] A.W.F. Huggins, R. Schwartz, R. Viswanathan and J. Makhoul, "Subjective Quality Testing of a New Source Model of LPC Vocoders," presented at the 96th meeting of the Acoustical Society of America, Nov. 1978, paper GGG12.

APPENDIX A

A CLASS OF ALL-ZERO LATTICE DIGITAL FILTERS: PROPERTIES AND APPLICATIONS

(Paper published in IEEE Trans. Acoustics, Speech and Signal Processing, pp. 304-314, Aug. 1978)

A Class of All-Zero Lattice Digital Filters: Properties and Applications

JOHN MAKHOUL, SENIOR MEMBER, IEEE

Abstract–**A class of minimum- or maximum-phase all-zero lattice digital filters, based on the two-multiplier lattice of Itakura and Saito, is developed. Different lattice forms with different numbers of multipliers are derived, including two one-multiplier forms. Many of the properties of these lattice filters are given, including the important orthogonalization and decoupling properties of successive stages in optimal inverse filtering of signals. These properties lead to important applications in the areas of adaptive linear prediction and adaptive Wiener filtering. As a specific example, the design of a new fast start-up equalizer is presented.**

Manuscript received September 29, 1977; revised March 9, 1978. This work was supported by the Information Processing Techniques Branch of the Advanced Research Projects Agency under Contract MDA903-75-C-0180.

The author is with Bolt Beranek and Newman, Inc., Cambridge, MA 02138.

I. Introduction

SEVERAL lattice and ladder structures have been proposed for the implementation of all-pole and pole-zero digital filters [1]-[5]. However, only a single lattice structure, due to Itakura and Saito [1], is available for the implementation of all-zero filters that are restricted to be minimum phase or maximum phase. This lattice filter structure has been useful in speech analysis applications [6] and promises to be useful in other areas as well, wherever transversal, predictive, or finite impulse response (FIR) filters are used in an adaptive manner.

The lattice of Itakura and Saito had two multipliers in each stage. In this paper, one-, two-, three-, and four-multiplier lattice structures will be developed for the implementation of minimum and maximum phase all-zero filters. Of particular interest is the one-multiplier form, because of the decreased number of multiplications. These structures are presented in Section IV. Before that, however, Section II presents the basic lattice of Itakura and Saito and develops some of its properties. More properties are derived in Section III where the application of the lattice to linear prediction is presented. Of importance are the orthogonalization and decoupling properties of the successive stages in the lattice. These properties are then used in Section V to show how the lattice can be employed very profitably in the areas of adaptive linear prediction and adaptive Wiener filtering. As a specific example, the design of a new and efficient fast start-up equalizer is presented.

One of the important properties of the lattice that is not discussed in this paper is its low sensitivity to roundoff noise [7], [8].

II. Basic All-Zero Lattice

Fig. 1 shows the basic two-multiplier lattice of Itakura and Saito [1], which they used for performing speech analysis. From Fig. 1, the following relations hold:

$$f_0(n) = g_0(n) = x(n), \tag{1a}$$

$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1), \tag{1b}$$

$$g_m(n) = K_m f_{m-1}(n) + g_{m-1}(n-1). \tag{1c}$$

$x(n)$ is the input signal, $f_m(n)$ is the "forward" residual at stage m, and $g_m(n)$ is the "backward" residual at stage m. In z-transform notation, (1) can be rewritten as

$$F_0(z) = G_0(z) = X(z), \tag{2a}$$

$$F_m(z) = F_{m-1}(z) + K_m z^{-1} G_{m-1}(z), \tag{2b}$$

$$G_m(z) = K_m F_{m-1}(z) + z^{-1} G_{m-1}(z). \tag{2c}$$

Let the forward and backward transfer functions at stage m be defined by

$$A_m(z) = \frac{F_m(z)}{X(z)} = \frac{F_m(z)}{F_0(z)} \tag{3a}$$

and

$$B_m(z) = \frac{G_m(z)}{X(z)} = \frac{G_m(z)}{G_0(z)}. \tag{3b}$$

Then, from (2) and (3) it is simple to see that $A_m(z)$ and $B_m(z)$ obey the recursion relations

$$A_0(z) = B_0(z) = 1, \tag{4a}$$

$$A_m(z) = A_{m-1}(z) + K_m z^{-1} B_{m-1}(z), \tag{4b}$$

$$B_m(z) = K_m A_{m-1}(z) + z^{-1} B_{m-1}(z). \tag{4c}$$

Furthermore, one can show from (4) that

$$B_m(z) = z^{-m} A_m(z^{-1}). \tag{5}$$

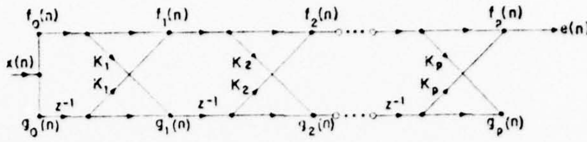


Fig. 1. The basic two-multiplier all-zero lattice of Itakura and Saito [1], [6].

Thus, if $A_m(z)$ is given by

$$A_m(z) = \sum_{k=0}^{m} a_m(k) z^{-k}, \tag{6a}$$

where $a_m(k)$ are the polynomial coefficients for an m-stage lattice, then

$$B_m(z) = \sum_{k=0}^{m} a_m(m-k) z^{-k}, \tag{6b}$$

and $B_m(z)$ is the reverse polynomial corresponding to $A_m(z)$.

From (4) and (6a) we also have

$$a_m(0) = 1,$$

$$a_m(m) = K_m. \tag{7}$$

Now, given some polynomial $A_p(z)$, with $a_p(0) = 1$, one can generate all the polynomials $A_m(z)$, $m < p$, and the coefficients K_m, using the following reverse recursion derived from (4):

$$K_m = a_m(m)$$

$$A_{m-1}(z) = \frac{A_m(z) - K_m B_m(z)}{1 - K_m^2}, \tag{8}$$

along with (5), and beginning with $m = p$. It is clear from (8) that should $|K_{m'}| = 1$ for some $m = m'$, then the solution for $A_{m'-1}(z)$ is indeterminate. Therefore, the reverse recursion (8) is possible iff $|K_m| \neq 1$, for all m.

It also follows from (5) and (6) that the zeros of $B_m(z)$ are the reciprocal of the zeros of $A_m(z)$. In particular, if all the zeros of $A_m(z)$ fall inside the unit circle, in which case $A_m(z)$ is minimum phase, then $B_m(z)$ is maximum phase. One can show that the minimum phase condition for $A_m(z)$ is guaranteed iff

$$-1 < K_i < 1, \quad 1 \leqslant i \leqslant m. \tag{9}$$

The coefficients K_m are then known as *reflection coefficients* or *partial correlation coefficients*. In much of the paper we shall assume that (9) holds, and therefore $A_m(z)$ and $B_m(z)$ are minimum phase and maximum phase, respectively.

A. Residual Energy

From (3a), the forward residual $f_m(n)$ can be represented by

$$F_m(z) = A_m(z) X(z). \tag{10}$$

By noting that the energy of $f_m(n)$ is equal to its zeroth autocorrelation coefficient, one can show, using (10), that the lattice structures will be developed for the implementation of minimum and maximum phase all-zero filters. Of particular interest is the one-multiplier form, because of the decreased number of multiplications. These structures are presented in Section IV. Before that, however, Section II presents the basic lattice of Itakura and Saito and develops some of its properties. More properties are derived in Section III where the application of the lattice to linear prediction is presented. Of importance are the orthogonalization and decoupling properties of the successive stages in the lattice. These properties are then used in Section V to show how the lattice can be employed very profitably in the areas of adaptive linear prediction and adaptive Wiener filtering. As a specific example, the design of a new and efficient fast start-up equalizer is presented.

One of the important properties of the lattice that is not discussed in this paper is its low sensitivity to roundoff noise [7], [8].

II. Basic All-Zero Lattice

Fig. 1 shows the basic two-multiplier lattice of Itakura and Saito [1], which they used for performing speech analysis. From Fig. 1, the following relations hold:

$$f_0(n) = g_0(n) = x(n), \tag{1a}$$

$$f_m(n) = f_{m-1}(n) + K_m g_{m-1}(n-1), \tag{1b}$$

$$g_m(n) = K_m f_{m-1}(n) + g_{m-1}(n-1). \tag{1c}$$

$x(n)$ is the input signal, $f_m(n)$ is the "forward" residual at stage m, and $g_m(n)$ is the "backward" residual at stage m. In z-transform notation, (1) can be rewritten as

$$F_0(z) = G_0(z) = X(z), \tag{2a}$$

$$F_m(z) = F_{m-1}(z) + K_m z^{-1} G_{m-1}(z), \tag{2b}$$

$$G_m(z) = K_m F_{m-1}(z) + z^{-1} G_{m-1}(z). \tag{2c}$$

Let the forward and backward transfer functions at stage m be defined by

$$A_m(z) = \frac{F_m(z)}{X(z)} = \frac{F_m(z)}{F_0(z)} \tag{3a}$$

and

$$B_m(z) = \frac{G_m(z)}{X(z)} = \frac{G_m(z)}{G_0(z)}. \tag{3b}$$

Then, from (2) and (3) it is simple to see that $A_m(z)$ and $B_m(z)$ obey the recursion relations

$$A_0(z) = B_0(z) = 1, \tag{4a}$$

$$A_m(z) = A_{m-1}(z) + K_m z^{-1} B_{m-1}(z), \tag{4b}$$

$$B_m(z) = K_m A_{m-1}(z) + z^{-1} B_{m-1}(z). \tag{4c}$$

Furthermore, one can show from (4) that

$$B_m(z) = z^{-m} A_m(z^{-1}). \tag{5}$$

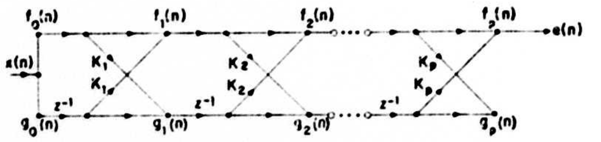


Fig. 1. The basic two-multiplier all-zero lattice of Itakura and Saito [1], [6].

Thus, if $A_m(z)$ is given by

$$A_m(z) = \sum_{k=0}^{m} a_m(k) z^{-k}, \tag{6a}$$

where $a_m(k)$ are the polynomial coefficients for an m-stage lattice, then

$$B_m(z) = \sum_{k=0}^{m} a_m(m-k) z^{-k}, \tag{6b}$$

and $B_m(z)$ is the reverse polynomial corresponding to $A_m(z)$.

From (4) and (6a) we also have

$$a_m(0) = 1,$$
$$a_m(m) = K_m. \tag{7}$$

Now, given some polynomial $A_p(z)$, with $a_p(0) = 1$, one can generate all the polynomials $A_m(z)$, $m < p$, and the coefficients K_m, using the following reverse recursion derived from (4):

$$K_m = a_m(m)$$
$$A_{m-1}(z) = \frac{A_m(z) - K_m B_m(z)}{1 - K_m^2}, \tag{8}$$

along with (5), and beginning with $m = p$. It is clear from (8) that should $|K_{m'}| = 1$ for some $m = m'$, then the solution for $A_{m'-1}(z)$ is indeterminate. Therefore, the reverse recursion (8) is possible iff $|K_m| \neq 1$, for all m.

It also follows from (5) and (6) that the zeros of $B_m(z)$ are the reciprocal of the zeros of $A_m(z)$. In particular, if all the zeros of $A_m(z)$ fall inside the unit circle, in which case $A_m(z)$ is minimum phase, then $B_m(z)$ is maximum phase. One can show that the minimum phase condition for $A_m(z)$ is guaranteed iff

$$-1 < K_i < 1, \quad 1 \leqslant i \leqslant m. \tag{9}$$

The coefficients K_m are then known as *reflection coefficients* or *partial correlation coefficients*. In much of the paper we shall assume that (9) holds, and therefore $A_m(z)$ and $B_m(z)$ are minimum phase and maximum phase, respectively.

A. Residual Energy

From (3a), the forward residual $f_m(n)$ can be represented by

$$F_m(z) = A_m(z) X(z). \tag{10}$$

By noting that the energy of $f_m(n)$ is equal to its zeroth autocorrelation coefficient, one can show, using (10), that the

$$C_p R_p^x C_p^T = E_p \tag{27}$$

where

$$C_p = \begin{bmatrix} 1 & & & & 0 \\ a_1(1) & 1 & & & \\ a_2(2) & a_2(1) & 1 & & \\ \cdot & \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & & \ddots \\ a_p(p) & a_p(p-1) & a_p(p-2) & \cdots & 1 \end{bmatrix} \tag{28}$$

and

$$E_p = \begin{bmatrix} E_0 & & & & 0 \\ & E_1 & & & \\ & & E_2 & & \\ & & & \ddots & \\ 0 & & & & E_p \end{bmatrix}. \tag{29}$$

Now, from (6b), we see that the rows of C_p are the coefficients of the backward filters $B_m(z)$. Therefore, using (3b), one can write

$$C_p x_p = g_p \tag{30a}$$

where

$$x_p = [x(n)x(n-1)\cdots x(n-p)]^T \tag{30b}$$

and

$$g_p = [g_0(n)g_1(n)\cdots g_p(n)]^T. \tag{30c}$$

The autocorrelation matrix for g_p is then given by

$$R_p^g = \overline{g_p g_p^T} = C_p \overline{x_p x_p^T} C_p^T$$
$$= C_p R_p^x C_p^T. \tag{31}$$

which, from (27), is equal to

$$R_p^g = E_p, \tag{32}$$

which is a diagonal matrix. Another way to write (32) is

$$\overline{g_i(n)g_j(n)} = \begin{cases} E_i, & i = j, \\ 0, & i \neq j. \end{cases} \tag{33}$$

This means that the backward residuals in the lattice are orthogonal to each other. Thus, the backward residuals are the result of an orthogonalization process (of the Gram-Schmidt type) on delayed versions of the signal $x(n)$. A different derivation of (33) as well as other correlation properties of the forward and backward signals are given in Appendix II.

The orthogonalization process results in a *decoupling* of the successive stages from each other. One of Itakura's contributions was to recognize this decoupling and use it to estimate the reflection coefficients in a straightforward manner. In fact, one can show that the global minimization of the output residual energy can be accomplished as a sequence of local minimization problems, one at each stage. Thus, one can obtain the optimal value of K_m that minimizes E_m by setting the derivative of E_m in (16) with respect to K_m to zero, and noting that E_{m-1} and r_{m-1} are not functions of K_m. The answer is

$$K_m^* = -r_{m-1}, \tag{34}$$

where r_{m-1} is the correlation coefficient in (15) between the two inputs to the mth stage. Substituting (34) in (16), the minimum residual energy at stage m is computed recursively from

$$E_m^* = (1 - K_m^{*2})E_{m-1}^*, \tag{35}$$

which gives the minimum residual energy at each stage in terms of the minimum residual energy of the previous stage and the reflection coefficient of the present stage. This is yet another manifestation of the decoupling between stages.

From (34) and (20) we conclude that K_m^* obeys (9), and therefore that the filter $A_p(z)$ that minimizes the output *residual energy is minimum phase*.

The minimum normalized residual energy at each stage m is obtained from (19a) and (35):

$$V_m^* = \prod_{i=1}^{m} (1 - K_i^{*2}), \tag{36}$$

where K_i^* is given by (34). From the discussion above, it is clear that V_m^* is bounded by

$$0 \leq V_m^* \leq 1. \tag{37}$$

Thus, while for an arbitrary filter V_m can have values greater than one, as in (21) and (22), using the filter that minimizes the output residual energy results in a V_m that is less than one.

Because of the equality of the energies of the forward and the backward residuals in (12), the correlation coefficients r_{m-1} in (15) can be computed in different ways. Another definition of r_{m-1}, and hence K_m^*, is obtained by minimizing *the sum of the forward and backward residual energies* at each stage. The answer, due to Burg [10], is

$$K_m^* = -r_{m-1} = -\frac{2\overline{f_{m-1}(n)g_{m-1}(n-1)}}{\overline{f_{m-1}^2(n)} + \overline{g_{m-1}^2(n-1)}}. \tag{38}$$

Both definitions (14) and (38) guarantee condition (9), even for a finite signal, and hence guarantee that $A_m(z)$ is minimum phase. Other definitions for K_m that guarantee (9) can be found in [11]. In the same reference, the author develops more efficient methods to compute (14) and (38).

The values of r_{m-1}, and therefore K_m^*, in (14) and (38) are equal if and only if $\overline{f_{m-1}^2(n)} = \overline{g_{m-1}^2(n-1)}$, which happens generally only if the signal is stationary or windowed. Different values result if the signal is not windowed [11], but both (14) and (38) continue to obey (9).

IV. All-Zero Lattice Realizations

A. *Alternate Forms*

In this section we shall develop several new all-zero lattice realizations in addition to Itakura's original realization. The resulting forms are canonic in the number of delays. Some of the realizations are actually in ladder form, but we shall use the word lattice to include both lattice and ladder forms. For each of the realizations only a single representative stage of the transfer function will be shown. The actual signal at each stage is obtained by multiplying the transfer function ($A_m(z)$ or $B_m(z)$) by the input $X(z)$.

Lattice Form 1 (LF1): Fig. 2 shows a stage of this lattice

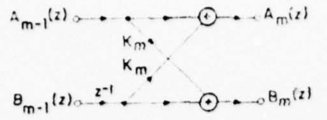


Fig. 2. The *m*th stage of lattice form 1 (LF1).

form, which is the form used in Fig. 1, and is represented by (4). The form has two multiplies and two adds. The addition nodes are shown explicitly to differentiate them from branch nodes.

Lattice Form 2 (LF2): Add and subtract $s_m K_m A_{m-1}(z)$ from (4b), and add and subtract $s_m K_m z^{-1} B_{m-1}(z)$ from (4c), where s_m is a sign parameter,

$$s_m = \pm 1, \quad \text{and} \quad s_m^2 = 1, \tag{39}$$

one obtains

$$A_m(z) = (1 - s_m K_m) A_{m-1}(z) + s_m K_m T_m(z) \tag{40}$$

$$B_m(z) = K_m T_m(z) + (1 - s_m K_m) z^{-1} B_{m-1}(z), \tag{41}$$

where

$$T_m(z) = A_{m-1}(z) + s_m z^{-1} B_{m-1}(z). \tag{42}$$

These equations represent form LF2, which can be shown to have three multiplies (neglecting the multiplies by - 1) and three adds. The choice of the sign of s_m for each stage is not important here. However, we shall see that the choice will be important in the one-multiplier realizations in Section IV-B.

Lattice Form 3 (LF3): LF3 is obtained similarly to LF2, but by adding and subtracting $s_m z^{-1} B_{m-1}(z)$ from (4b), and adding and subtracting $s_m A_{m-1}(z)$ from (4c). The result is

$$A_m(z) = T_m(z) + (K_m - s_m) z^{-1} B_{m-1}(z) \tag{43}$$

$$B_m(z) = (K_m - s_m) A_{m-1}(z) + s_m T_m(z) \tag{44}$$

where $T_m(z)$ is given by (42). LF3 uses two multiplies and three adds.

By substituting for $A_{m-1}(z)$ from (4b) in (4c) one obtains a different form with three multiplies and two adds. A similar form is also obtained by substituting for $z^{-1} B_{m-1}(z)$ from (4c) in (4b). The details are left to the reader.

Although the additional lattice forms derived above are interesting variations on the basic lattice, they do not seem to offer any particular advantages over LF1; in fact, they all require more computations, either in the number of multiplications or the number of additions or both. However, by appropriate manipulation it is possible to transform LF2 and LF3 to forms with only a single multiplier. This is described below.

B. One-Multiplier Forms

The lattice transfer function may be scaled by multiplying each stage by some multiplier M_m, $1 \leqslant m \leqslant p$. The scaled transfer functions are then given by

$$\begin{aligned} \hat{A}_p(z) &= P_p A_p(z) \\ \hat{B}_p(z) &= P_p B_p(z) \end{aligned} \tag{45}$$

where

$$P_p = \prod_{m=1}^{p} M_m. \tag{46}$$

Equations (40)-(42) for LF2 are then transformed to

$$\begin{aligned} \hat{A}_m(z) &= M_m [(1 - s_m K_m) \hat{A}_{m-1}(z) + s_m K_m \hat{T}_m(z)] \\ \hat{B}_m(z) &= M_m \{K_m \hat{T}_m(z) + (1 - s_m K_m) z^{-1} \hat{B}_{m-1}(z)\} \end{aligned} \tag{47}$$

where

$$\hat{T}_m(z) = \hat{A}_{m-1}(z) + s_m z^{-1} \hat{B}_{m-1}(z). \tag{48}$$

By setting

$$M_m = \frac{1}{1 - s_m K_m} \tag{49}$$

(47) becomes

$$\begin{aligned} \hat{A}_m(z) &= \hat{A}_{m-1}(z) + s_m \frac{K_m}{1 - s_m K_m} \hat{T}_m(z) \\ \hat{B}_m(z) &= \frac{K_m}{1 - s_m K_m} \hat{T}_m(z) + z^{-1} \hat{B}_{m-1}(z). \end{aligned} \tag{50}$$

The flow diagrams for (50) are shown in Fig. 3(a) for $s_m = +1$, and in Fig. 3(b) for $s_m = -1$. This form, which we shall label LF2/1, has only a single multiply but the three adds remain. Thus the total number of operations is equal to that of LF1, except that one multiply has been replaced by one add. Where multiplication is expensive computationally, or for hardware implementation, LF2/1 can result in substantial savings.

LF3 can be transformed similarly using the multiplier

$$M_m = \frac{1}{K_m - s_m} \tag{51}$$

with (43) and (44). The result is

$$\begin{aligned} \hat{A}_m(z) &= \frac{1}{K_m - s_m} \hat{T}_m(z) + z^{-1} \hat{B}_{m-1}(z) \\ \hat{B}_m(z) &= \hat{A}_{m-1}(z) + s_m \frac{1}{K_m - s_m} \hat{T}_m(z). \end{aligned} \tag{52}$$

The flow graphs are shown in Fig. 4 for $s_m = \pm 1$, and we shall call this form LF3/1. It also has one multiply and three adds.

Residual Energy: The residual energy at stage m for the transformed lattice is given from (11) and (45) by

$$\hat{E}_m = P_m^2 \prod_{i=-m}^{m} a_m(i) R(i). \tag{53}$$

and from (18) and (46) by

$$\hat{E}_m = R(0) \prod_{i=1}^{m} M_i^2 (1 + 2 r_{i-1} K_i + K_i^2). \tag{54}$$

In the context of linear prediction analysis, the *minimum residual energy* for the transformed lattice is given from (36) and (46) by

$$\hat{E}_m^* = R(0) \prod_{i=1}^{m} M_i^2 (1 - K_i^{*2}). \tag{55}$$

For LF2/1, we have from (39), (49) and (55)

$$\hat{E}_m^* = R(0) \prod_{i=1}^{m} \frac{1 + s_i K_i^*}{1 - s_i K_i^*}. \tag{56}$$

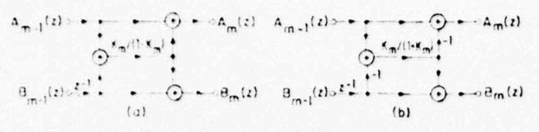


Fig. 3. One-multiplier lattice form LF2/1. (a) $s_m = +1$, (b) $s_m = -1$.

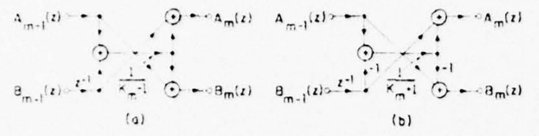


Fig. 4. One-multiplier lattice form LF3/1. (a) $s_m = +1$, (b) $s_m = -1$.

One can also show that the minimum residual energy for LF3/1 is given by (56) as well, since $(1 - s_m K_m)^2 = (K_m - s_m)^2$. The minimum normalized residual energy $\hat{V}^*_m$ in each case can be obtained simply by dividing (55) or (56) by $R(0)$.

Choice of Sign Parameters: One can choose the sign parameters to suit any particular application. In finite wordlength (FWL) computations, one is often interested in minimizing the scaling operations to make maximum use of the number of bits available. One reasonable criterion to meet [4] is to have the energy at each point in the lattice be as constant as possible, and in particular not to exceed some specified overflow value, say T. At each stage compute $\hat{E}_m$ from

$$\hat{E}_m = \frac{1 + 2r_{m-1}K_m + K_m^2}{(1 - s_m K_m)^2}\,\hat{E}_{m-1} \tag{57}$$

for $s_m = +1$ and $s_m = -1$, and choose the value of s_m that results in $\hat{E}_m$ being closer to but not exceeding the threshold T. For an optimal inverse filter (one that minimizes the residual energy), the residual energy is computed from

$$\hat{E}^*_m = \frac{1 + s_m K^*_m}{1 - s_m K^*_m}\,\hat{E}^*_{m-1}. \tag{58}$$

Since, in this case, $|K^*_m| < 1$, one can increase or decrease E^*_m relative to E^*_{m-1} by proper choice of s_m. Thus, for $s_m = \operatorname{sgn} K^*_m$

$$\frac{\hat{E}^*_m}{\hat{E}^*_{m-1}} = \frac{1 + |K^*_m|}{1 - |K^*_m|} > 1 \tag{59}$$

and for $s_m = -\operatorname{sgn} K^*_m$

$$\frac{\hat{E}^*_m}{\hat{E}^*_{m-1}} = \frac{1 - |K^*_m|}{1 + |K^*_m|} < 1. \tag{60}$$

C. Normalized Forms

By specifying appropriate multipliers M_m at each stage, one can ensure that the normalized energy at each stage is unity [7]. The resulting forms shall be called *normalized* forms. Below, we shall assume that the filter is the optimal inverse filter and, hence, that the minimum residual energy is given by (55). The normalized minimum energy is then

$$\hat{V}_m = \prod_{i=1}^{m} M_i^2 (1 - K_i^2), \tag{61}$$

where we have dropped the asterisks for convenience. It is clear from (61) that $\hat{V}_m$ can be made to equal unity by setting

$$M_i = \frac{1}{\sqrt{1 - K_i^2}} \tag{62}$$

at each stage. Introducing (62) in lattice forms 1-3 results in the corresponding normalized forms. Below, we shall give the results for lattice forms 1 and 2 only; the others can be derived in a similar fashion.

Normalized Lattice Form 1 (NLF1): Multiplying (4) by M_m in (62), one obtains

$$\begin{aligned}\hat{A}_m(z) &= \frac{1}{\sqrt{1 - K_m^2}}\,\hat{A}_{m-1}(z) + \frac{K_m}{\sqrt{1 - K_m^2}}\,z^{-1}\hat{B}_{m-1}(z)\\ \hat{B}_m(z) &= \frac{K_m}{\sqrt{1 - K_m^2}}\,\hat{A}_{m-1}(z) + \frac{1}{\sqrt{1 - K_m^2}}\,z^{-1}\hat{B}_{m-1}(z).\end{aligned} \tag{63}$$

By setting

$$K_m = \cos w_m \tag{64}$$

where w_m is an angle between $-\pi$ and π, (63) reduces to

$$\begin{aligned}\hat{A}_m(z) &= \csc w_m \hat{A}_{m-1}(z) + \operatorname{ctn} w_m z^{-1}\hat{B}_{m-1}(z)\\ \hat{B}_m(z) &= \operatorname{ctn} w_m \hat{A}_{m-1}(z) + \csc w_m z^{-1}\hat{B}_{m-1}(z).\end{aligned} \tag{65}$$

The resulting normalized form is shown in Fig. 5.

Normalized Lattice Form 2 (NLF2): Substituting for M_m from (62) into (47), there results

$$\begin{aligned}\hat{A}_m(z) &= \sqrt{\frac{1 - s_m K_m}{1 + s_m K_m}}\,\hat{A}_{m-1}(z) + s_m \frac{K_m}{\sqrt{1 - K_m^2}}\,\hat{T}_m(z)\\ \hat{B}_m(z) &= \frac{K_m}{\sqrt{1 - K_m^2}}\,\hat{T}_m(z) + \sqrt{\frac{1 - s_m K_m}{1 + s_m K_m}}\,z^{-1}\hat{B}_{m-1}(z)\end{aligned} \tag{66}$$

which, by substituting (64), can be shown to reduce to

$$\begin{aligned}\hat{A}_m(z) &= \left(\tan\frac{w_m}{2}\right)^{s_m}\hat{A}_{m-1}(z) + s_m \operatorname{ctn} w_m \hat{T}_m(z)\\ \hat{B}_m(z) &= \operatorname{ctn} w_m \hat{T}_m(z) + \left(\tan\frac{w_m}{2}\right)^{s_m} z^{-1}\hat{B}_{m-1}(z).\end{aligned} \tag{67}$$

The flow graphs for $s_m = +1$ and -1 are shown in Fig. 6.

Finally, we note that in the case where the lattice is the optimal inverse filter, the backward residuals in the normalized structures are orthonormal to within a multiplicative constant:

$$\overline{\hat{g}_i(n)\hat{g}_j(n)} = \begin{cases} R(0), & i = j,\\ 0, & i \neq j.\end{cases} \tag{68}$$

V. Applications

In this section we present two applications of all-zero lattice digital filters: 1) adaptive linear prediction; and 2) adaptive Wiener filtering in the form of an efficient fast start-up equalizer. In both applications one can use any of the struc-

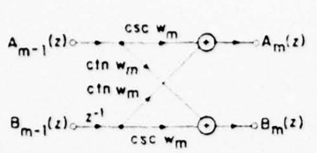


Fig. 5. Normalized lattice form NLF1.

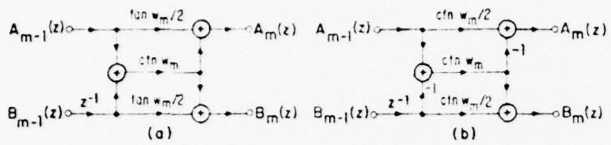


Fig. 6. Normalized lattice form NLF2. (a) $s_m = +1$, (b) $s_m = -1$.

tures given in this paper, but the one-multiplier structures are of special interest because of the reduced number of multiplies.

A. Adaptive Linear Prediction

Adaptive linear prediction or adaptive inverse filtering has been useful in many applications such as speech, radar and sonar processing, and adaptive noise canceling (see, for example, [12]). In those applications it has been customary to use a transversal filter to perform the filtering. However, with some increase in computation one can use the all-zero lattice to give more reliable, stable, and less noisy estimates of the filter coefficients. Below, we describe briefly one method for adaptively computing the reflection coefficients; other methods are given elsewhere [11], [13], [14].

Given $K_m(n)$, $1 \leqslant m \leqslant p$, at time n, and the forward and backward signals up to time n, we shall compute $K_m(n+1)$ using the following realization of (38):

$$K_m(n+1) = -\frac{2\sum_{i=i_0}^{n} \beta^{n-i} f_{m-1}(i) g_{m-1}(i-1)}{\sum_{i=i_0}^{n} \beta^{n-i}[f_{m-1}^2(i) + g_{m-1}^2(i-1)]} \tag{69}$$

$$= -\frac{C(n)}{D(n)} \tag{70}$$

where

$$0 < \beta \leqslant 1. \tag{71}$$

The lower limit i_0 in the summations in (69) depends on the type of estimator memory to be used. We differentiate two types of memory:

(1) Growing memory (i_0 is a fixed integer, such as zero);

(2) Fixed memory ($i_0 = n - M + 1$, where M is the fixed memory size).

Also, $\beta = 1$ represents a nonfading memory, and $\beta < 1$ a fading memory. It is common to use either a fixed-nonfading memory or a growing-fading memory.

Note that with the definition in (69), $|K_m(n)| < 1$ for all m and n. Other definitions which contain the energy of either the forward or the backward residual but not both, require fewer computations but do not guarantee (9) to hold [11]. However, for some adaptive applications, (9) need not hold [15].

Many adaptive applications use a growing-fading memory, i.e., i_0 is fixed and $\beta < 1$. For this case, we have from (69) and (70) the following recursive relations:

$$C(n) = \beta C(n-1) + 2 f_{m-1}(n) g_{m-1}(n-1) \tag{72a}$$

$$D(n) = \beta D(n-1) + f_{m-1}^2(n) + g_{m-1}^2(n-1). \tag{72b}$$

The adaptive procedure, then, operates as follows. Given the forward and backward signals at time n, use (72) and (70) to compute $K_m(n+1)$ for $1 \leqslant m \leqslant p$. Using the new values of the reflection coefficients and any of the lattice structures given in Section IV, compute the forward and backward signals at time $n+1$. The process is then repeated.

The fading or attenuation factor β determines the "effective" memory size of the estimator. When β is close to 1 the adaption is slow, and is fast for small values of β. In fact, it is clear from (72) that the effect of β is that of a single-pole low-pass filter operating on $C(n)$ and $D(n)$; β is the value of the pole in the z-plane.

Equations (69)-(72) can be used with any of the lattice structures in this paper, including the one-multiplier structures. In the latter case, one may compute the lattice multiplier value directly from $C(n)$ and $D(n)$, without first computing K_m. For example, the multiplier of Fig. 3(a) can be computed as follows:

$$\frac{K_m}{1-K_m} = -\frac{C(n)}{D(n)+C(n)}.$$

For the special case of Itakura's two-multiplier structure in Fig. 1, one can show by simple manipulation that (69) can be written in the form

$$K_m(n+1) = K_m(n) - \frac{f_{m-1}(n) g_m(n) + g_{m-1}(n-1) f_m(n)}{D(n)}. \tag{73}$$

where $D(n)$ is given recursively by (72b). Equation (73) says that each coefficient at time $n+1$ is obtained by adding a correction term to the value at time n. One can see, for example, that for a growing-nonfading memory ($\beta = 1$), $D(n)$ increases continuously and, therefore, the correction term in (73) tends to zero as n goes to infinity. In this case, K_m tends to its optimal value with probability 1, assuming a stationary signal. It is interesting to note that (73) represents essentially a steepest descent gradient algorithm for estimating K_m. For $\beta < 1$, and appropriate normalization, one can show that (73) is similar to the lattice estimate of Griffiths [16], and becomes equal to it in the steady state with a gradient step size equal to $1 - \beta$. What we have shown here essentially is that, due to the equivalence of (73) to (69), the estimate in (73) is always guaranteed to obey (9).

As a result of the orthogonalization and decoupling properties of the lattice, the convergence of the adaptive lattice is much faster than that for the corresponding adaptive transversal filter. Griffiths [16] has noted that the convergence time of the lattice is almost independent of the eigenvalue spread of the signal, i.e., independent of its spectral dynamic range.

B. A Fast Start-Up Equalizer

As an application of lattice filters to adaptive Wiener filtering, we present the design of a new fast start-up equalizer.

During a start-up period prior to data transmission, the tap coefficients of an adaptive equalizer are adjusted automatically to optimum values that minimize the distortion or mean-square error of the received pulses. It is desirable that the start-up time used for this initial adaption process be as small as possible. Chang [17] proposed an equalizer structure that reduces the start-up time drastically. The general form of the structure is shown in Fig. 7. The tap coefficients c_i are adjusted such that the mean-square error between $y(n)$ and some reference signal is minimized. If the filters are selected such that the signals $z_i(n)$ are orthonormal, then the tap coefficients c_i can be adjusted to their optimum values in one step [17].

The specific fast start-up equalizer proposed by Chang is shown in Fig. 8. The filter signals $z_i(n)$ are obtained from $x(n)$ by a linear transformation

$$z = Px \tag{74}$$

where

$$z = [z_1(n) z_2(n) \cdots z_N(n)]^T, \tag{75}$$

x is given by (30b) with p replaced by $N - 1$, and P is an $N \times N$ transformation matrix to be determined.

To enable fast start-up, the matrix P is chosen such that [17]

$$PRP^T = I, \tag{76}$$

where $R = R^x$ is the autocorrelation matrix (order $N - 1$) of the signal $x(n)$ [see (26)], and I is the unit diagonal matrix. The signal $x(n)$ is taken here to be the response of the channel to an impulse or a pseudorandom sequence. The solution chosen for P by Chang is

$$P = D^{-1/2} Q^T \tag{77}$$

where

$$R = QDQ^T, \tag{78}$$

Q is a matrix whose columns are the orthonormal eigenvectors of R, and D is a diagonal matrix whose elements are the eigenvalues of R. From (77) and (78), and noting that $Q^{-1} = Q^T$, it is clear that (76) holds and that the signals $z_i(n)$ defined by (74) and (75) are orthonormal.

The structure in Fig. 8 requires N^2 coefficients with an equal number of multiplies. This number can grow rapidly as the number of stages N in the equalizer is increased. Using special properties of the matrix R, Cantoni and Butler [18] showed that the same equalizer structure can be implemented using $N^2/2 + N$ coefficients, a saving of about one-half. However, the number of coefficients still increases as the square of the number of stages. Below, we shall show that, using a lattice structure for the fast start-up part of the equalizer, the number of coefficients becomes a linear function of N.

Let

$$P = E_{N-1}^{-1/2} C_{N-1} \tag{79}$$

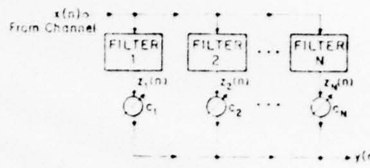


Fig. 7. Generalized equalizer structure.

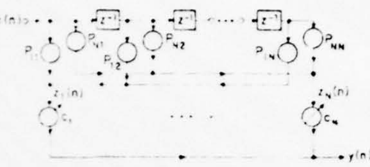


Fig. 8. Fast start-up equalizer structure of Chang [17].

where C and E are given by (28) and (29), respectively. By using (27), it is simple to show that P in (79) obeys (76), as desired. By applying P in (79) to the input x, it is clear from (74) and (30) that z can be obtained by taking the orthogonal signals $g_m(n)$ along the backward path of the lattice and multiplying them by $E_m^{-1/2}$ to normalize them.

To ensure fast start-up, one can show that (76) need hold only to within a multiplicative constant. Thus, instead of choosing P as in (79) we could choose P to equal

$$P = V_{N-1}^{-1/2} C_{N-1} \tag{80}$$

where the elements of V are the normalized residual energies:

$$V = \frac{1}{R(0)} E. \tag{81}$$

We then have from (80), (81) and (27):

$$PRP^T = R(0) I. \tag{82}$$

Therefore, the backward signals $g_m(n)$ are multiplied by $V_m^{-1/2}$ to normalize them to within a constant equal to the signal energy. This constant can then easily be incorporated into the adjustment of the coefficients c_i of the equalizer proper.

Fig. 9(a) shows the general lattice structure for the proposed equalizer, and Fig. 9(b) shows the structure when the lattice uses one of the normalized forms in Figs. 5 and 6. The values of V_m to use must correspond to the particular lattice form chosen. For example, if one uses the two-multiplier form shown in Figs. 1 and 2, then V_m is given by (36), or if the one-multiplier forms shown in Figs. 3 and 4 are employed, then V_m as derived from (56) is used. On the other hand, when the normalized forms are used, V_m is unity, as shown in Fig. 9(b).

The total number of coefficients employed with either the two-multiplier forms or the three-multiplier normalized forms in Fig. 6 is $3(N - 1)$. The minimum number of coefficients is achieved by the one-multiplier forms, and is equal to $2(N - 1)$. Therefore, the number of coefficients in the equalizer is now linear with the number of stages, which makes equalizers with a large number of stages much cheaper to implement. Furthermore, from the discussion in Section V-A, it becomes clear that the lattice equalizer should adapt to new channels in a simple and efficient manner.

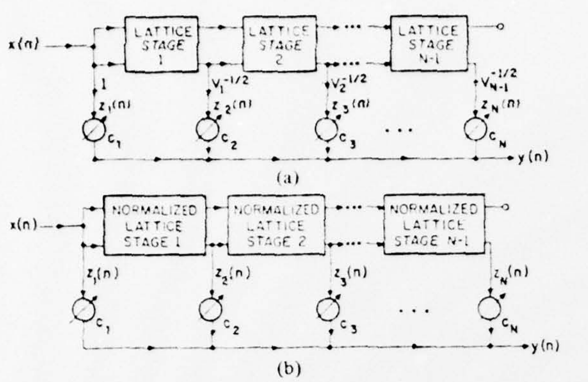


Fig. 9. (a) General lattice fast start-up equalizer structure. (b) Lattice fast start-up equalizer using normalized lattice forms.

C. Discussion

From (76), one can write

$$R^{-1} = P^T P. \tag{83}$$

Therefore, the problem of choosing a transformation P that renders the signals $z_i(n)$ orthonormal, can be seen from (83) to be a problem in the factorization of the inverse of a covariance matrix. Assuming R to be positive definite, which is generally true for physical signals, there exists an infinity of possible solutions for P. Two important factorizations were given above. The first was an eigenvector factorization, where the orthogonal decomposition of x into z is often termed a Karhunen-Loève decomposition. The second factorization is of the LDL^T type, where L is lower triangular with 1's along the main diagonal. This factorization can be shown to be equivalent to a Gram-Schmidt orthogonalization of x into z. The triangular decomposition results in a matrix P with $N(N+1)/2$ elements, as compared to N^2 in the eigenvector decomposition, and hence offers a reduction of about one-half in the number of coefficients. In addition, it offers substantial savings in computation due to the fact that triangular factorization is much simpler and less expensive than computing eigenvalues and eigenvectors. However, the major benefit accrues when R is a symmetric Toeplitz matrix. For this special but important case, the triangular decomposition can be implemented in a lattice form, with the number of coefficients being linear with N. What the lattice effectively does is perform the Gram-Schmidt orthogonalization recursively; each stage does its best in decorrelating the two inputs entering it. There are cases where R is not Toeplitz, but where a good suboptimal lattice solution may still be found (see [11]).

Finally, we point out that the structures in Fig. 9 can be used in place of any adaptive transversal Wiener filter in order to speed up the convergence of parameter estimation.

VI. Conclusion

Based on the two-multiplier all-zero lattice of Itakura and Saito, a number of other lattice forms were derived. Of those forms, perhaps the one-multiplier is of greatest interest, because it is canonical in the number of multiplies.

Properties of the all-zero lattice were given in some detail. The specific simultaneous implementation of the minimum-phase and maximum-phase filters in the lattice leads to interesting properties, including the important orthogonalization and decoupling properties. These properties make the lattice especially attractive in adaptive linear prediction, or in other areas of Wiener filtering where transversal or FIR filters are used in an adaptive manner. The given adaptive algorithms for the estimation of the reflection coefficients is seen to be superior to similar algorithms using a transversal filter because of the inherent stability of the lattice structure, the fast convergence of lattice algorithms, and the relative independence of the convergence with respect to the signal eigenvalue spread.

We also showed how the lattice can be used in the efficient design of a fast start-up equalizer. The number of coefficients used can be as few as $2N - 1$, where N is the number of equalizer taps, compared to $N^2/2 + N$ using the equalizer structure of Chang, and Cantoni and Butler.

Appendix I

We wish to show that (27) holds, i.e.,

$$CRC^T = E \tag{27}$$

where we have dropped the subscripts and superscripts for clarity. Because R is symmetric about both major diagonals, one can rewrite (26a) as

$$R \begin{bmatrix} a_p(p) \\ \cdot \\ \cdot \\ a_p(1) \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \cdot \\ \cdot \\ 0 \\ E_p \end{bmatrix}. \tag{A-1}$$

This equation applies for any value of p, and therefore

$$R \begin{bmatrix} 1 & a_1(1) & a_2(2) \cdots a_p(p) \\ & 1 & a_2(1) \cdots a_p(p-1) \\ & & 1 \quad \cdots a_p(p-2) \\ & & \quad \cdot \quad\quad \cdot \\ & & \quad\quad \cdot \quad \cdot \\ 0 & & \quad\quad\quad 1 \end{bmatrix}$$

$$= RC^T = \begin{bmatrix} E_0 & & & 0 \\ & E_1 & & \\ & & E_2 & \\ & & & \ddots \\ X & & & E_p \end{bmatrix}, \tag{A-2}$$

where the lower triangular part X is possibly nonzero. It then follows that

$$[RC^T]^T = CR^T = CR \tag{A-3}$$

is upper triangular. Now, since both CR and C^T are upper triangular, their product is upper triangular and the elements along the main diagonal is the product of the diagonal elements from both matrices. Therefore,

$$CRC^T = \begin{bmatrix} E_0 & & & Y \\ & E_1 & & \\ & & \ddots & \\ 0 & & & E_p \end{bmatrix}. \tag{A-4}$$

Since $[CRC^T]^T = CRC^T$, the product matrix is symmetric, and the part denoted by Y in (A-4) must be zero. Therefore,

$$CRC^T = \begin{bmatrix} E_0 & & & 0 \\ & E_1 & & \\ & & \ddots & \\ 0 & & & E_p \end{bmatrix} = E. \tag{A-5}$$

Appendix II

Correlation Properties

Below, we present some properties of the correlations between the forward and backward residuals in the lattice.

From (3) and (6), we have

$$f_m(n) = \sum_{k=0}^{m} a_m(k)x(n-k) \tag{B-1a}$$

$$g_m(n) = \sum_{k=0}^{m} a_m(m-k)x(n-k) \tag{B-1b}$$

with

$$a_m(0) = 1, \tag{B-2b}$$

$$a_m(m) = K_m, \tag{B-2b}$$

as in (7). We shall assume in the sequel that the predictor coefficients $a_m(k)$ are those that minimize the energy in the forward and backward residuals, and hence obey (24) and (25), which we rewrite here as

$$\sum_{k=0}^{m} a_m(k)R(i-k) = 0, \qquad 1 \leqslant i \leqslant m, \tag{B-3}$$

$$\sum_{k=0}^{m} a_m(k)R(k) = E_m, \tag{B-4}$$

where E_m is the minimum energy at stage m, and $R(k)$ is the autocorrelation of the signal $x(n)$ defined by

$$\overline{x(n-i)x(n-j)} = R(j-i). \tag{B-5}$$

Most of the properties given below make simple use of the equations above. Hence, no detailed derivations are given.

A. Properties

The first two properties are restatements of the orthogonality condition (B-3):

$$\overline{f_m(n)x(n-i)} = 0, \qquad 1 \leqslant i \leqslant m. \tag{B-6}$$

$$\overline{g_m(n)x(n-i)} = 0, \qquad 0 \leqslant i \leqslant m-1. \tag{B-7}$$

Note the difference between the ranges of i in (B-6) and (B-7). Using (B-4), we have

$$\overline{f_m(n)x(n)} = \overline{g_m(n)x(n-m)} = E_m. \tag{B-8}$$

Now, making use of (B-6)-(B-8) also, we have the following two properties:

$$\overline{f_i(n)f_j(n)} = E_{\max(i,j)} \tag{B-9}$$

$$\overline{g_i(n)g_j(n)} = E_i\delta_{ij}, \tag{B-10}$$

where

$$\delta_{ij} = \begin{cases} 1, & i=j \\ 0, & i \neq j. \end{cases}$$

Equation (B-10) is identical to (33). Note from (B-9) that the forward residuals do not exhibit the same orthogonality property as the backward residuals. However, there is a certain duality between the forward and backward residuals, first exhibited in (B-1) and then in (B-8). We shall see now that (B-9) and (B-10) also have duals.

From (B-6) and (B-7), one can establish the following conditions for the orthogonality between delayed versions of the residuals:

$$\overline{f_i(n)f_j(n-t)} = 0 \quad \text{for} \begin{cases} 1 \leqslant t \leqslant i-j, & i>j \\ -1 \geqslant t \geqslant i-j, & i<j \end{cases} \tag{B-11}$$

$$\overline{g_i(n)g_j(n-t)} = 0 \quad \text{for} \begin{cases} 0 \leqslant t \leqslant i-j-1, & i>j \\ 0 \geqslant t \geqslant i-j+1, & i<j \end{cases} \tag{B-12}$$

where t is an integer lag. The only value of t in (B-12) that is the same for $i>j$ and $i<j$ is $t=0$. For $t=0$, we already have (B-9) and (B-10). The duals stem from (B-11), where the only value of t that is the same for $i>j$ and $i<j$ is $t = i-j$. Making use of the general relation $\overline{v(n)w(n-t)} = \overline{v(n+t)w(n)}$, we have for $t = i-j$

$$\overline{f_i(n+i)f_j(n+j)} = E_i\delta_{ij} \tag{B-13}$$

$$\overline{g_i(n+i)g_j(n+j)} = E_{\max(i,j)}. \tag{B-14}$$

Equations (B-13) and (B-14) are the duals of (B-9) and (B-10). (B-13) is the orthogonality equation for the forward residuals.

Making use of (B-2b) one can derive the following cross-correlation property:

$$\overline{f_i(n)g_j(n)} = \begin{cases} K_jE_i, & i \geqslant j, \\ 0, & i<j, \end{cases} \quad i,j \geqslant 0 \tag{B-15}$$

where we have assumed that $K_0 = 1$.

From (15) and (34), we have

$$\overline{f_i(n)g_i(n-1)} = -K_{i+1}E_i. \tag{B-16}$$

From (B-16) one can show that

$$\overline{g_i(n-1)x(n)} = \overline{f_i(n+1)x(n-i)} = -K_{i+1}E_i. \tag{B-17}$$

In general, we have the following cross-correlation property:

$$\overline{f_i(n)g_j(n-1)} = \begin{cases} 0, & i > j, \\ -K_{j+1}E_j, & i \leqslant j, \end{cases} \quad i, j \geqslant 0. \tag{B-18}$$

Note the difference between (B-15) and (B-18).

ACKNOWLEDGMENT

The author wishes to thank L. J. Griffiths of the University of Colorado, E. Satorius of the Naval Ocean Systems Center, and R. Viswanathan, H. Gish, and T. Fortmann of Bolt Beranek and Newman, Inc., for helpful discussions relating to this paper.

REFERENCES

[1] F. Itakura and S. Saito, "Digital filtering techniques for speech analysis and synthesis," in *Proc. 7th Int. Cong. Acoust.*, Budapest, 1971, Paper 25-C-1, pp. 261-264.

[2] A. Fettweis, "Some principles of designing digital filters imitating classical filter structures," *IEEE Trans. Circuit Theory*, vol. CT-18, pp. 314-316, Mar. 1971.

[3] S. K. Mitra and R. J. Sherwood, "Digital ladder networks," *IEEE Trans. Audio Electroacoust.*, vol. AU-21, pp. 30-36, Feb. 1973.

[4] A. H. Gray, Jr., and J. D. Markel, "Digital lattice and ladder filter synthesis," *IEEE Trans. Audio Electroacoust.*, vol. AU-21, pp. 491-500, Dec. 1973.

[5] S. K. Mitra, P. S. Kamat, and D. C. Huey, "Cascaded lattice realization of digital filters," *Circuit Theory and Applications*, vol. 5, pp. 3-11, 1977.

[6] F. Itakura and S. Saito, "On the optimum quantization of feature parameters in the PARCOR speech synthesizer," in *Proc. IEEE Conf. Speech Commun. Processing*, Newton, MA, 1972, pp. 434-437.

[7] J. D. Markel and A. H. Gray, Jr., "Roundoff noise characteristics of a class of orthogonal polynomial structures," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 473-486, Oct. 1975.

[8] C. T. Mullis and R. A. Roberts, "Roundoff noise in digital filters: Frequency transformations and invariants," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, pp. 538-550, Dec. 1976.

[9] J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, pp. 561-580, Apr. 1975.

[10] J. P. Burg, "Maximum entropy spectral analysis," Ph.D. dissertation, Dep. Geophysics, Stanford Univ., Stanford, CA, May 1975.

[11] J. Makhoul, "Stable and efficient lattice methods for linear prediction," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, pp. 423-428, Oct. 1977.

[12] B. Widrow, J. McCool, M. Larimore, and C. R. Johnson, Jr., "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proc. IEEE*, vol. 64, pp. 1151-1162, Aug. 1976.

[13] R. Viswanathan and J. Makhoul, "Sequential lattice methods for stable linear prediction," presented at EASCON '76, Washington, DC, paper 155.

[14] J. Makhoul and R. Viswanathan, "Adaptive lattice methods for linear prediction," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Tulsa, OK, Apr. 1978, pp. 83-86.

[15] L. J. Griffiths, "Rapid measurement of digital instantaneous frequency," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 207-222, Apr. 1975.

[16] ——, "A continuously-adaptive filter implemented as a lattice structure," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Hartford, CT, May 1977, pp. 683-686.

[17] R. W. Chang, "A new equalizer structure for fast start-up digital communication," *Bell Syst. Tech. J.*, vol. 50, pp. 1969-2014, July-Aug. 1971.

[18] A. Cantoni and P. Butler, "Properties of the eigenvectors of persymmetric matrices with applications to communication theory," *IEEE Trans. Comm.*, vol. COM-24, pp. 804-809, Aug. 1976.

APPENDIX B

A MIXED-SOURCE MODEL FOR SPEECH COMPRESSION AND SYNTHESIS

(Paper published in J. Acoust. Soc. Am., Vol.64, No. 6, pp. 1577-1581, Dec. 1978.)

A mixed-source model for speech compression and synthesis

John Makhoul, R. Viswanathan, Richard Schwartz, and A. W. F. Huggins

Bolt Beranek and Newman Incorporated, Cambridge, Massachusetts 02138

(Received 12 May 1978; revised 30 August 1978)

This paper presents an excitation source model for speech compression and synthesis that allows the *degree* of voicing to be varied continuously by mixing voiced (pulse) and unvoiced (noise) excitations in a frequency-selective manner. The mix is achieved by dividing the speech spectrum into two regions, with the pulse source exciting the low-frequency region and the noise source exciting the high-frequency region. The degree of voicing is specified by a parameter F_c, which corresponds to the cut-off frequency between the voiced and unvoiced regions. For speech compression applications, F_c can be extracted automatically from the speech spectrum and transmitted. Experiments performed with the new model indicate its power in synthesizing natural sounding voiced fricatives and in largely eliminating the "buzzy" quality of vocoded speech. A functional definition of buzziness and naturalness is given in terms of the model.

PACS numbers: 43.70.Lw, 43.70.Jt

INTRODUCTION

Perhaps the single most important decision to be made in a pitch-excited speech compression system (vocoder) is the voiced/unvoiced (V/U) decision. Errors made in this decision will readily be perceived by the ear as a degradation of speech quality, which may also be accompanied by a loss in intelligibility. Yet, even if the V/U decision were somehow to be made "perfectly," the resynthesized speech would still lack naturalness, because of various other factors. In this paper, we shall deal with those aspects of naturalness that can be influenced by the source excitation, particularly "buzziness" and "lack of fullness"—two qualtities often associated with vocoded speech, especially for speech generated by a linear prediction (LPC) vocoder.

There are, of course, different types of buzziness caused by different factors, not all of which are related to the source excitation. In the work described in this paper, however, we have investigated the degree to which buzziness may be reduced by source manipulation, regardless of the cause of the buzziness.

In this paper, we explore the excitation problem in speech synthesis and present a simple mixed-source model that allows for a *degree* of voicing. The new model is capable of producing more natural-sounding speech; it seems to eliminate most of the problem of buzziness and recover part of the fullness of natural speech. In addition, the model promises to reduce the adverse effects of voicing errors. Other mixed-source models have been used previously in speech compression and synthesis (Holmes, 1973; Fujimura, 1966, 1968; Itakura and Saito, 1968; Kato *et al.*, 1967; Coulter, 1975; Klatt, 1976; Strube, 1977). A review of previous research relating to the model presented in this paper is given in Sec. VII.

I. BASIC SYNTHESIS MODEL AND TERMINOLOGY

Throughout this paper, we shall assume the basic synthesis model shown in Fig. 1. In this model, a time-varying excitation signal excites a time-varying spectral shaping filter to produce the synthetic speech. The excitation signal is assumed to have a *flat spectrum*, so that the spectral envelope of the synthetic speech is determined completely by the spectral shaping filter. Furthermore, we shall assume this model to hold for any type of synthesis, whether as part of a vocoder system or a speech synthesis system. In fact, we will argue that our proposed source model is indeed adequate for both applications.

Assuming, then, that the excitation has a flat spectrum, we are necessarily limited to two types of excitation: deterministic (pulse) or random (noise).

A. Pulse source (buzz)

The deterministic excitation used is, in general, the impulse response of an all-pass filter, which we shall call an all-pass signal or pulse. The simplest form of an all-pass pulse is a single impulse. When the pulse source produces a sequence of pulses separated by a pitch period, it is known as a *buzz* source. (Note that a single pulse could be used in the synthesis of the burst in a plosive sound (Holmes, 1973). However, the burst can also be synthesized using the noise source. We shall assume the latter in this paper; the pulse source will be used exclusively for buzz excitation.)

B. Noise source (hiss)

The random noise excitation may be the output of a random number generator. Generators with either a uniform or a Gaussian probability distribution are readily available and appear to be quite adequate. The noise source is also known as a *hiss* source.

Whether the actual excitation is buzz, hiss, or a combination of the two, it is important that the excitation have a flat spectrum. We next describe how one might derive an appropriate source model by inspecting short-time speech spectra.

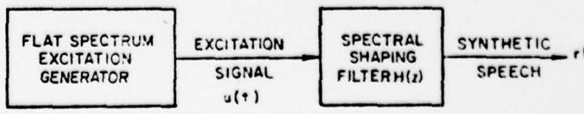

FLAT SPECTRUM EXCITATION GENERATOR
EXCITATION SIGNAL
u(t)
SPECTRAL SHAPING FILTER H(z)
SYNTHETIC SPEECH
r(t)

FIG. 1. Basic synthesis model.

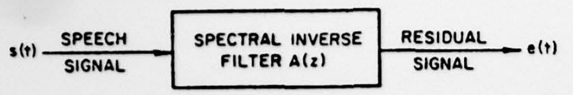


FIG. 2. Inverse filtering the speech signal to obtain a residual signal with a flat spectrum.

II. THE "IDEAL" SOURCE

For some particular speech signal, one can remove the short-time spectral envelope by appropriate inverse filtering, as shown in Fig. 2. The inverse filter $A(z)$ can be obtained by cepstral techniques (Oppenheim and Schafer, 1975) or through the use of linear prediction (Makhoul, 1975). The residual signal $e(t)$ will then have a nominally flat spectrum. If, in Fig. 1, the excitation $u(t)$ can be made identical to the residual $e(t)$ and the synthesis filter $H(z)$ is the inverse of $A(z)$, the reconstructed synthetic speech $r(t)$ will be identical to the original input signal $s(t)$.

However, for general synthesis purposes, the synthetic signal need only *sound* like the original; it need not be identical to it. In addition, in synthesis we need to manipulate the source pitch, and in vocoding we need to minimize the number of bits required to represent the source. To accomplish these tasks, we make use of an important property of speech perception: its relative insensitivity to the short-time phase. Therefore, to model the residual $e(t)$ to meet our requirements, we need only look at its spectrum and, except for pitch, disregard its phase structure for the moment.

Figure 3 shows the signal power spectrum of 25.6 ms of a 10-kHz sampled signal in the middle of the vowel [I] in the word "list," and the corresponding residual spectrum. The residual was obtained by inverse filtering the speech signal with a 20th-order linear prediction inverse filter. If one could generate an excitation $u(t)$ whose spectrum is identical to the residual spectrum, the synthetic speech might then sound almost the same as the original.

Therefore, our aim in developing a source model is to obtain an excitation spectrum that is as close as possible to the residual spectrum. Furthermore, in obtaining such a spectrum, we want to use only the buzz and hiss sources described in Sec. I. The source model will follow naturally from the characteristics of residual spectra.

III. CHARACTERISTICS OF RESIDUAL SPECTRA

The residual spectrum in Fig. 3 shows a clear periodicity up to about 3.5 kHz and a lack of periodicity above that frequency. The periodicity is shown by the regularly spaced peaks in the spectrum that correspond to the harmonics of the voice fundamental frequency. Inspection of residual spectra of other sounds shows that the existence of aperiodic frequency bands in sonorant sounds is quite common. While only two bands can be identified in Fig. 3, one periodic and one aperiodic, it is possible to have several adjacent periodic and aperiodic bands in 5 kHz. For more examples, the reader is referred to the work of Fujimura (1968).

Partial devoicing of certain sounds is well known from physical considerations. For example, [z] is devoiced above about 1 kHz, and several attempts at the synthesis of more natural voiced fricatives have made use of this fact. On the other hand, it is also known that in the production of the tense front vowel [i], the constriction may become narrow enough to generate some turbulence, which results in devoicing of frequencies above about 3 kHz. To date, however, most synthesizers have not taken advantage of this fact.

In addition to the foregoing examples of devoicing, Fujimura (1968) has hypothesized that devoicing of some spectral regions may be due to aperiodicities or irregularities in vocal-cord movement. We have observed that spectral devoicing often occurs during transitions between different sounds, including sonorant–sonorant transitions. In contrast to the examples given in the previous paragraph, we believe that the spectral devoicing due to vocal-cord irregularities and/or spectral transitions may in fact be an artifact of the spectral estimation process. It is not clear whether it is appropriate to use a noise source for synthesizing such devoiced regions.

In conclusion, residual spectra may be completely periodic (voiced), completely aperiodic (unvoiced), or may contain some spectral regions that are periodic and others that are aperiodic. We discuss next how such spectra can best be modeled using the buzz and hiss sources.

IV. PROPOSED SOURCE MODEL

One reasonable source model would divide the spectrum into a number of bands. Each band considered to be periodic would be excited by the buzz source, and each band considered to be aperiodic would be excited by the hiss source. Fujimura (1968) used a three-band model in his experiment with a channel vocoder, and reported an improvement in speech naturalness. However, given our observations that spectral aperiodicities may not necessarily result from turbulent excitations, we have chosen a simpler model. We treat as periodic all

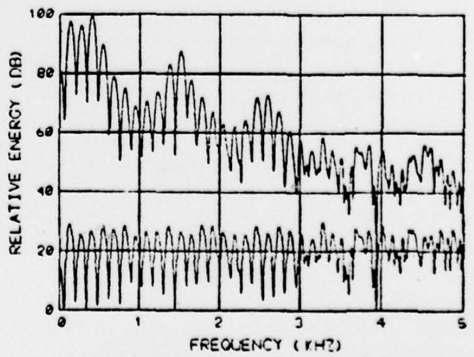


FIG. 3. Signal spectrum (top) and residual spectrum (bottom) for the vowel [I] in the word "list."

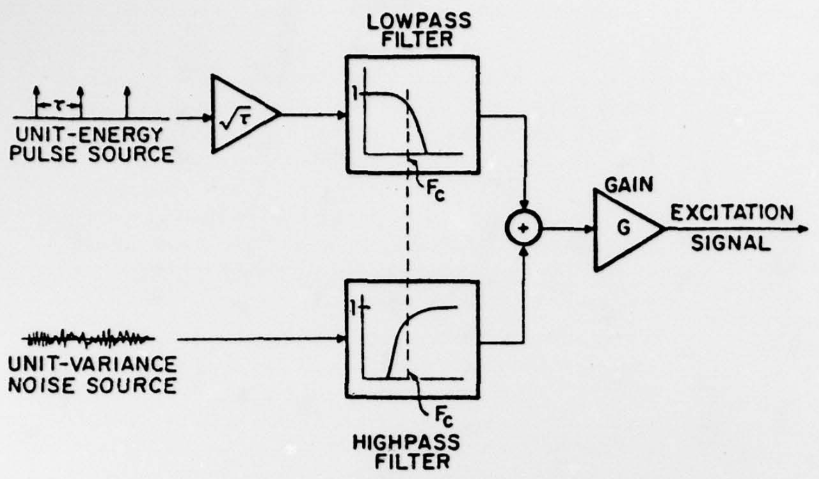


FIG. 4. Frequency-selective mixed-source excitation model.

spectral aperiodic regions that are in between two periodic regions. In other words, only the band above the highest frequency region that is periodic will be treated as aperiodic and will be excited by a turbulent source. There are two reasons for this choice: (a) Turbulent sources are more likely to excite high frequencies than low, and (b) excessive devoicing can degrade quality just as severely as excessive voicing.

The resulting model is shown in Fig. 4. It is a mixed-source model where the buzz source excites a time-varying low-frequency region of the spectrum, and the hiss source excites the remaining high-frequency region. The excitation signal is realized by passing the pulse excitation through a low-pass filter with cutoff F_c, and the noise excitation through a high-pass filter with the same cutoff frequency F_c, then adding the outputs of the two filters. The resulting mixed-source excitation signal is multiplied by the source gain and applied to the spectral shaping filter. The model, then, has only two parameters: the cutoff frequency F_c, and the pitch period τ when $F_c > 0$. Since small changes in F_c do not seem to be perceptible, it is sufficient to quantize F_c into 2–3 bits for transmission purposes.

V. IMPLEMENTATION

A. Extraction of source parameters

The only novelty in parameter extraction for the new source model is that the traditional binary V/U decision has been replaced by the determination of a multivalued parameter F_c. The extraction of the pitch period is not affected, and we shall not address that problem here.

The question, of course, is how to compute F_c in a perceptually satisfactory manner. One of the algorithms we had considered but rejected, computed F_c from the autocorrelation of the residual. We had hoped that the normalized autocorrelation of the residual, at a lag equal to the pitch period, could be calibrated to indicate the degree of voicing. However, it became clear that the value of the autocorrelation at the pitch period was complicated by the interaction between the data window and the pitch period. Therefore, we abandoned this line of attack in favor of the method described below.

The method we have chosen thus far is a peak-picking algorithm on the signal spectrum. The algorithm determines periodic regions of the spectrum by examining the separation between consecutive peaks and determining whether the separations are the same, within some tolerance level. F_c is taken to be the highest frequency at which the spectrum is considered to be periodic.

For each speech frame, we obtain a pitch frequency estimate using a modified version of a recently developed harmonic pitch extractor (Seneff, 1977), which also employs peak picking of the signal spectrum and therefore fits very naturally into our scheme. Using this pitch frequency estimate, the algorithm tests whether the separation between adjacent peaks are within some tolerance from the pitch estimate. F_c is then that frequency beyond which the separation between peaks do not fall within the given tolerance levels. The algorithm includes heuristics that take into account occasional regions of aperiodicity as well as shifts in pitch frequency from one spectral region to another. (The latter phenomenon might be attributable to the changing pitch frequency within one analysis frame.)

After extracting F_c for consecutive frames, a three-point median smoother is applied to the computed cutoff frequency values. This smoothing technique corrects the cutoff frequency upward whenever it is lower than two adjacent values and, therefore, eliminates spurious low cutoff frequency values.

B. Filter implementations

In our initial implementation, we rounded the value of F_c to the nearest 500 Hz. Therefore, we needed lowpass and highpass filters with cutoff frequencies separated by 500 Hz. The filter designs were then stored and used in the synthesis as the need arose.

For each value of F_c, the 3-dB points for the lowpass and highpass filters were designed to be equal to F_c, so that the spectrum of the final excitation may be as flat as possible. The rolloff of the filters was considered to be of secondary importance; filters with transition regions of about 1000 Hz were felt to be suitable. We considered FIR (finite impulse response) as well as recursive (low order Butterworth) filters. The two types of filters gave similar perceptual results.

VI. RESULTS

Using the implementation described in Sec. V, we compared the resulting synthesized speech to speech that employed the binary V/U model in the context of a linear prediction (LPC) vocoder. A number of sentences from male and female speakers (Huggins, Viswanathan, and Makhoul, 1977) were used in comparing the two analysis–synthesis systems. No quantization of parameters (except for F_c) was performed. One of the sentences had a concentration of fricative sounds: "His vicious father has seizures"; and another was a nonnasal sonorant sentence: "Why were you away a year, Roy?" Other sentences were more general. With the V/U source, the fricative sentence sounded particularly buzzy for both male and female speakers, while the sonorant sentence was judged as buzzy only for low-pitched male speakers. The buzziness in both sentences was greatly reduced when using the mixed-source model. In general, the buzziness was always reduced with the new model. However, for some sentences the new synthesis produced certain small background noises. Upon careful listening, it was determined that some of these noises were also present in the V/U synthesis, but were masked by the buzziness. The other noises may be due to inaccurate determination of F_c and/or to the particular implementation of the model.

Overall, listeners thought that the new model performed better for female speakers. The new synthesis was "raspier" and more characteristic of female speech, which is considered to be more breathy than male speech. A number of listeners reported that the new synthesis had a certain "fullness" that was absent with the V/U synthesis. We interpret this as an indication of the greater naturalness resulting from the new model.

Formal testing of our implementation of the mixed-source model is planned. The results will be reported in a subsequent paper.

VII. REVIEW OF RELATED WORK

We know of only one other work in which mixed excitation was used with LPC vocoders: that of Itakura and Saito (1968). There, however, the two sources excited the whole spectrum simultaneously, with the "degree" of voicing being controlled by the relative *amplitudes* of the sources. The results were not encouraging (Itakura).

After the development of our model over two years ago, we became aware of Fujimura's work (1966, 1968). As far as we know, he was the first to suggest and test a frequency-selective mixed-source model. His work, which we mentioned earlier, was performed in the context of a pitch-excited channel vocoder. Fujimura brought to our attention his other work with Kato *et al.* (1967), which employed a variable cutoff frequency like ours, but used a different algorithm to determine the cutoff. The work was done with a hybrid voice-excited and pitch-excited channel vocoder, and the researchers reported excellent results. Cutler (1975) used mixed excitation for the synthesis of voiced fricatives; however, the cutoff between the low- and high-frequency bands was fixed.

In speech synthesis, mixed excitation has been used routinely for the synthesis of voiced obstruents (see, for example, Holmes, 1973; Klatt, 1976). The parallel formant synthesizer of Holmes (1973) allows for variable mixed excitation, and was especially used in transitions between unvoiced and voiced sounds. Upon careful reading, it became clear to us that the spirit of Holmes' synthesizer is similar to ours, except that, in his case, the controls are more complicated. A more recent hardware synthesizer by Strube (1977) allows for mixed excitation using a single variable RC circuit.

There have been numerous attempts at reducing buzziness by changing the shape of the pulse in voiced excitation, and results have been mixed. Recently, Sambur *et al.* (1977) reported a reduction in vowel buzziness (exhibited mainly with low-pitch voices) by changing the pulse width to be proportional to the pitch period. Unfortunately, changing the pulse width changes the excitation spectrum; the effect is that of a variable lowpass filter. Spectrally flattening the pulse before excitation seemed to cancel the reduction in buzziness (Atal, 1977).

VIII. DISCUSSION

A. Buzziness and naturalness

It is interesting that the mixed-source model appears to reduce two seemingly different types of buzziness: the buzziness in voiced fricative synthesis and the buzziness in sonorant synthesis associated mainly with low-pitched voices. Our hypothesis is that the two types of buzziness may, in fact, result from the same process: that of an excess in buzz source excitation. Thus, our general rule is that:

Too much buzz results in "buzziness."

Too much hiss results in "breathiness" or "raspiness."

If more of the spectrum is excited by the buzz source than is necessary for naturalness, the result is buzziness. Similarly, if there is more hiss excitation than is necessary for naturalness, the result is breathiness or raspiness. These observations lead us to a functional definition of one aspect of naturalness, as it relates to mixed excitation:

> Naturalness is maximized by that proper mix of buzz and hiss excitations that leads to a synthesis that is neither buzzy nor breathy (or raspy).

We must emphasize here that the mixed-source model may not remove all types of buzziness. In fact, it is conceivable that for certain types of buzziness, such as in sonorant sounds, the mixed source may be simply masking the buzziness by substituting hiss for buzz at high frequencies. That such a solution seems to succeed in eliminating the buzziness does not necessarily mean that other solutions may not be more appropriate.

B. Modulation and naturalness

Certain synthesizers, such as that of Klatt (1976), amplitude-modulate the hiss source by the buzz source for the synthesis of voiced fricatives. While it is known that the noise source in the vocal tract is in fact modulated by the vocal cord output, it is not clear that such modulation is necessary for achieving naturalness in synthetic speech. Whatever effect modulation has, it appears to be of a secondary nature. Holmes (1973) reported very natural speech for his synthesizer, which did not contain any modulation. Although we initially included modulation in our model, it is at present our opinion that source modulation may not be necessary for natural synthesis, and therefore we have decided not to incorporate it as part of the model.

C. Phase and naturalness

It is generally agreed that proper phase determination of buzz excitation should lead to more natural synthesis. Furthermore, such phase cannot be in the form of some "optimal" pitch pulse shape. The phase must change from one pitch pulse to the next in some appropriate manner. Thus far, our model calls for an all-pass pulse, but does not specify the phase. Exactly how the phase should change between pulses is a subject for future research.

IX. CONCLUSION

We have presented a frequency-selective mixed-source excitation model for use in *both* speech compression and speech synthesis. The model has a *single* continuous parameter, F_c, which effectively divides the spectrum into two regions. A buzz source excites the low-frequency region below F_c, and the hiss source excites the high-frequency region above F_c. Naturalness (no buzziness or breathiness) is maximized by the proper mix of the two sources, i.e., by the proper determination of F_c. For sentences where F_c was properly extracted, the speech sounded much more natural. Improper determination of F_c may lead to either buzziness or breathiness.

While other aspects of the synthesis process may be more important in producing greater naturalness, we believe that any model that attempts to achieve naturalness must include in it some type of mixed excitation. The mixed-source model presented in this paper is a step in that direction.

ACKNOWLEDGMENTS

The authors wish to thank K. N. Stevens for many discussions, especially during the initial development of the model. One of the authors (JM) had a useful discussion with L. Rabiner concerning the different types of buzziness in vocoded speech. This work was sponsored by the Information Processing Techniques Branch of the Advanced Research Projects Agency under Contract No. MDA903-75-C-0180.

Atal, B. (1977). Personal communication.
Coulter, D. (1975). "Application of Simultaneous Voice/Unvoice Excitation in a Channel Vocoder," U. S. Patent No. 3,903,366.
Fujimura, O. (1966). "Speech Coding and the Excitation Signal," 1966 IEEE Int. Comm. Conf., Digest Tech. Pap., p. 49.
Fujimura, O. (1968). "An Approximation to Voice Aperiodicity," IEEE Trans. Audio Electroacoust. AU-16, 68–72 (March).
Holmes, J. N. (1973). "The Influence of Glottal Waveform on the Naturalness of Speech from a Parallel Formant Synthesizer," IEEE Trans. Audio Electroacoust. AU-21, 298–305 (June).
Huggins, A. W. F., Viswanathan, R., and Makhoul, J. (1977). " Speech-quality testing of some variable-frame-rate (VFR) linear-predictive (LPC) vocoders," J. Acoust. Soc. Am. 62, 430–434.
Itakura, F., and Saito, S. (1968). "Analysis Synthesis Telephony Based upon the Maximum Likelihood Method," Proc. 6th Int. Congr. Acoust., Tokyo, Japan, Paper C-5-5, pp. C17-20.
Itakura, F. Personal communication.
Kato, Y., Ochiai, K., Fujimura, O., and Maeda, S. (1967). "A Vocoder Excitation with Dynamically Controlled Voicedness," 1967 Conf. Speech Comm. and Processing, Cambridge, MA, pp. 288–291.
Klatt, D. H. (1976). "Structure of a Phonological Rule Component for a Synthesis-by-Rule Program," IEEE Trans. Acoust. Speech Signal Process. ASSP-24, 391–398 (October).
Makhoul, J. (1975). "Linear Prediction: A Tutorial Review," Proc. IEEE 67, 561–580 (April).
Oppenheim, A. V., and Schafer, R. W. (1975). *Digital Signal Processing* (Prentice-Hall, Englewood Cliffs).
Sambur, M., Rosenberg, A., Rabiner, L., and McGonegal, C. (1977). "On Reducing the Buzz in LPC Synthesis," 1977 IEEE Int. Conf. Acoust. Speech Signal Process., Hartford, CT, pp. 401–404.
Seneff, S. (1977). "Real Time Harmonic Pitch Detector," Report 1977-5, Lincoln Lab., MIT, Lexington, MA (January).
Strube, H. W. (1977). "Synthesis Part of a 'Log Area Ratio' Vocoder in Analog Hardware," IEEE Trans. Acoust. Speech Signal Process., ASSP-25, 387–391 (October).

APPENDIX C

HIGH-FREQUENCY REGENERATION IN SPEECH CODING SYSTEMS

(Presentation given at the IEEE Int. Conf. Acoustics, Speech and Signal Processing, Washington, DC, pp. 428-431, April 1979.)

HIGH-FREQUENCY REGENERATION IN SPEECH CODING SYSTEMS

John Makhoul and Michael Berouti

Bolt Beranek and Newman Inc.
Cambridge, MA 02138

ABSTRACT

The traditional method of high-frequency regeneration (HFR) of the excitation signal in baseband coders has been to rectify the transmitted baseband, followed by spectral flattening. In addition, a noise source is added at high frequencies to compensate for lack of energy during certain sounds. In this paper, we reexamine the whole HFR process. We show that the degree of rectification does not affect the output speech, and that, with proper processing, the high-frequency noise source may be eliminated. We introduce a new type of HFR based on spectral duplication of the baseband. Two types of spectral duplication are presented: spectral folding and spectral translation. Finally, in order to eliminate the problem of breaking the harmonic structure due to spectral duplication, we propose a pitch-adaptive spectral duplication scheme in the frequency domain by using adaptive transform coding to code the baseband.

1. INTRODUCTION

Baseband coders, or what are known also as voice-excited coders [1-8], were originally proposed as a compromise between pitch-excited coders (such as LPC, channel and homomorphic vocoders) and waveform coders. Today, baseband coders offer attractive alternatives at data rates in the range 6.4-9.6 kb/s. This range of data rates has become increasingly important because modems are now available that operate reliably in that range over regular telephone lines.

Below, we shall assume that, at the receiver, the synthesizer obeys the general synthesis model shown in Fig. 1. In the figure, the synthetic or reconstructed speech signal r(t) is generated as the result of applying a time-varying excitation signal u(t) as input to a time-varying spectral shaping filter. The spectral envelope of the excitation is assumed to be flat, so that the spectral envelope of the synthetic speech is determined completely by the spectral shaping filter. The parameters of the model, i.e., the excitation and the filter, must be computed and transmitted periodically by the transmitter. Those parameters that represent the speech spectrum, denoted as spectral parameters, are computed, quantized and transmitted every 15-30 ms. In a baseband coder, a low-frequency portion of the excitation, known as a <u>baseband</u>, is transmitted and used at the receiver to regenerate the high-frequency portion of the excitation. The sum of the transmitted baseband and the regenerated high-frequency band constitute the excitation u(t) to the synthesizer.

In a baseband coder, synthetic speech quality is determined by four factors: a) width of the baseband, b) coding of the baseband, c) estimation and coding of spectral parameters, and d) the high-frequency regeneration (HFR) method employed. In this paper we shall concentrate mainly on the fourth factor, HFR.

2. BASEBAND CODERS

Fig. 2 shows the transmitter portion of a digital baseband coder, based on a linear prediction representation. In this and other figures, we indicate the Nyquist frequency (half the sampling frequency) in parentheses at each point in the process. The speech signal s(t), sampled at 2W Hz, is first filtered with the LPC inverse filter A(z) to produce the residual e(t). (The parameters of A(z) are transmitted separately.) Baseband extraction is usually in the form of a lowpass or bandpass filter of width B Hz. The signal b(t) is known as the baseband residual, which is decimated to a signal v(t) sampled at 2B Hz. v(t) is then coded and transmitted.

The coding may be quite simple, using adaptive quantization, or may be more complicated, employing adaptive predictive coding (APC), adaptive transform coding (ATC), or sub-band coding (SBC). Only APC has been used previously in coding the baseband residual [8], while SBC has been used in coding a baseband of the speech signal itself [7]. In Section 6 we propose the use of ATC in baseband residual coding.

Fig. 3 shows a block diagram of a typical receiver for a baseband coder. The difference between the interpolated signal b'(t) and the baseband residual b(t) should be primarily due to quantization and coding. The signal b'(t) goes through a HFR process and a highpass filter that keeps only the high frequencies not contained in b'(t). The sum of the resulting signal and b'(t) form u(t), the excitation to the all-pole LPC synthesizer, which produces the output speech.

3. HIGH-FREQUENCY REGENERATION

It is well-known that if the baseband has either the voice fundamental or at least two adjacent harmonics, a waveform containing all the harmonics of voiced input speech can be generated by feeding the baseband signal to an instantaneous, zero-memory, nonlinear device. The spectral shape of the regenerated harmonic structure may be quite arbitrary and must be flattened to produce a suitable excitation function.

A digital implementation of a general HFR system is shown in Fig. 4. High frequencies are generated by performing some form of nonlinear distortion on the baseband signal. Because such distortion may introduce energy at frequencies higher than W, it is recommended that the baseband be interpolated to at least double the original sampling rate before the distortion, in order to avoid spectral aliasing [7] which can cause roughness in the output speech. The distorted signal is then spectrally flattened before it is used as excitation to the synthesizer. In most systems to date, a noise source is added to the distorted signal to compensate for the loss of high frequencies in fricatives. However, we have found that if the spectral flattening is performed in some optimal way (using LPC, for example), the noise source is unnecessary.

Rectification

Most nonlinear distortion schemes to date use some form of waveform rectification. In general, a rectifier operating on a signal x(t) has the following input-output characteristic

$$y(t) = [(1+\alpha)|x(t)| + (1-\alpha)x(t)]/2 \quad (1)$$

where $|\cdot|$ denotes absolute value, and α is some constant in the range $0 \leq \alpha \leq 1$. $\alpha=0$ corresponds to half-wave rectification and $\alpha=1$ corresponds to full-wave rectification. Both of these values have been used in the past, as well as a value of $\alpha=0.5$ [6]. We now show that the value of α should have no effect on the output speech.

We assume the signal x(t) to be bandlimited to B Hz; hence, x(t) has no energy above B Hz. The full-wave rectified signal |x(t)| will have energy at frequencies above B Hz. It is clear from (1), then, that the high-frequency energy in y(t) above B Hz is completely determined by that in |x(t)|. Except for a scaling factor, the spectra of y(t) and |x(t)| are identical above B Hz. The spectral shape of y(t) below B Hz depends on the value of α. However, we note from Fig. 3 that only the region above B Hz is taken from the output of HFR. Therefore, we conclude that α does not affect the shape of the spectrum of the excitation u(t) and consequently does not affect the output speech. This conclusion has been borne out in experiments. The user is then free to choose the type of rectification based on ease of implementation, without the choice having an effect on the output speech.

Spectral Flattening

Schroeder et al. [1-3] used waveform clipping to perform spectral flattening. More recently, Un and Magill [5] used double differencing to emphasize high frequencies, while Weinstein [6] used LPC spectral flattening. On the other hand, Esteban et al. [7] did not report any spectral flattening in their system.

We have used adaptive LPC spectral flattening in conjunction with full-wave rectification to implement a 9.6 kb/s baseband coder. Although the output speech was of high quality, a small amount of roughness could be perceived upon listening with a headset.

4. HFR BY SPECTRAL DUPLICATION

Here we present a new HFR method based on duplication of the baseband spectrum. The idea behind the new method derives from the pitch-excited coder. In voiced excitation, the spectrum of the excitation is a flat line spectrum at multiples of the fundamental pitch frequency. Such a spectral structure is periodic and repetitive: the high-frequency structure is the same as at low frequencies. The spectrum of unvoiced excitation, on the other hand, is continuous and has a random spectrum with a flat envelope. However, the details of the unvoiced spectrum are not as perceptually important as the details of the voiced spectrum. Therefore, the unvoiced spectrum can be considered repetitive also, in that any similar spectrum can be substituted with equally good results.

The new regeneration method, then, is simply to duplicate the baseband spectrum at higher frequencies, in some fashion. We shall show systems that perform spectral duplication in each of two ways: a) spectral folding, and b) spectral translation. Below, we shall assume that the signal bandwidth W is an integer multiple of the baseband width B, and we shall denote W/B=L. This integer-band assumption greatly simplifies the two implementations.

Fig. 5 shows the desired results for L=3. Fig. 5a shows the baseband residual spectrum; Fig. 5b shows the result of spectral folding, and Fig. 5c the result of spectral translation. In Fig. 5b, the spectrum in the second band (between B and 2B) is the mirror image (folded version) of the baseband, while the spectrum in the third band is a folded version of the spectrum in the second band, which makes the spectrum in the third band identical in shape to the baseband spectrum. In Fig. 5c, the second and third bands have spectra identical to the baseband. One can think of the higher bands being obtained as a result of translating the baseband.

Spectral Folding

Fig. 6 shows the complete receiver using integer-band spectral folding. To perform an L-band spectral folding, one simply inserts L-1 zeros between samples of the transmitted baseband. This process is merely that of upsampling, which is well known to produce spectral folding automatically. Therefore, essentially no computations are needed to generate the excitation. Because the baseband residual has a flat spectrum, the excitation will have a flat spectrum, and there is no need for spectral flattening.

Spectral Translation

Fig. 7 shows the general system for integer-band spectral translation. Note that multiplication by $(-1)^t$ gives a signal with the mirror image spectrum. Again in Fig. 7 we use upsampling to produce spectral folding. The filter H(z) is a multiple bandpass filter, as shown in Fig. 8 for L=3, which passes those bands that have the same shape as the baseband, i.e., every other band. The filter 1-H(z) then passes the intervening bands. The sum of the outputs of the two filters constitutes the excitation for the synthesizer. The computations implied in Fig. 7 may be reduced substantially by making use of appropriate symmetries.

5. INITIAL RESULTS

In preliminary experiments using HFR by spectral folding we heard a number of distortions in the form of added tones. These tones were generally more audible with a larger number of bands (i.e., larger L) and for higher-pitched voices. We were able to verify the existence of a tone at even multiples of the folding frequency, i.e., at multiples of 2B Hz. The tone was largely eliminated by a simple method: we subtracted off the short-term d.c. in the baseband signal, because the d.c. is folded into multiples of 2B Hz. After spectral folding, the original d.c. was restored so as not to disturb the average signal level, but the energy at multiples of 2B Hz had already been eliminated.

Other audible tones have been more difficult to trace. However, we are currently working on this problem. We are also experimenting with the alternative spectral translation method.

Although spectral folding seems to generate certain background low-level tones, it does not seem to have any perceptible roughness, as was the case in rectification. We see this difference as a tradeoff between the two methods at this time.

One reason for the existence of these background tones may be the fact that, with spectral duplication, the harmonic structure is interrupted at multiples of B Hz. Therefore, one could hypothesize that the tones may be eliminated by adjusting the width B of the baseband to be a multiple of the pitch fundamental frequency on a short-term basis. Unfortunately, such a scheme would require an enormous amount of computation, which would offset the initial reduction in computation afforded by the spectral duplication method. However, if somehow one could perform the baseband coding in the frequency domain instead of the time domain, pitch-adaptive spectral duplication could be accomplished very easily. This is the basis for our proposed system in the next section.

6. BASEBAND ADAPTIVE TRANSFORM CODER

The idea here is to use ATC to code the baseband residual. In ATC, the time signal is transformed to another domain, quantized and coded in that domain. If the discrete cosine transform is used, then the coding is in the frequency domain.

In addition to the usual analysis at the transmitter, one also extracts the pitch for each frame. At the receiver, the baseband frequency components are duplicated at higher frequencies. The folding or translation frequency can be easily changed each frame to be a multiple of the pitch fundamental frequency. Note that the fact that the signal bandwidth will not be an integer multiple of the now pitch-adaptive baseband, poses no problems at all. Therefore, it makes this method very general.

7. CONCLUSIONS

We discussed the problem of high-frequency regeneration (HFR) in baseband coders. We showed that all forms of rectification are equivalent in their performance as HFR agents. When adaptive LPC is used to perform spectral flattening on the rectified baseband, it was found that the addition of extra noise at high frequencies was not necessary. However, a small amount of roughness was perceived at 9.6 kb/s.

Two forms of spectral duplication, spectral folding and spectral translation, were suggested as new HFR methods. Initial results showed the existence of background tones, some of which have been successfully eliminated. As a way to remedy the break in the harmonic structure produced by spectral duplication, we proposed a pitch-adaptive baseband coding system using adaptive transform coding with spectral duplication in the frequency domain.

ACKNOWLEDGEMENTS

The authors wish to thank R. Viswanathan, W. Russell, R. Schwartz and D. Green for fruitful discussions relating to this work. This work was supported by the Advanced Research Projects Agency and monitored by RADC/ETC under contract number F19628-78-C-0136.

REFERENCES

1. M.R. Schroeder, "Recent Progress in Speech Coding at Bell Telephone Laboratories," in Proc. IIIrd Int. Cong. Acoust., Stuttgart, Germany, 1959.
2. M.R. Schroeder and E.E. David, Jr., "A Vocoder for Transmitting 10 kc/s speech over a 3.5 kc/s channel," Acustica, Vol. 10, pp. 35-43, 1960.
3. E.E. David, M.R. Schroeder, B.F. Logan, and B.F. Prestigiacomo, "New Applications of Voice-excitation of Vocoders," in Proc. Stockholm Speech Comm. Seminar, Royal Inst. Tech., Stockholm, Sweden, 1962.
4. B. Gold and J. Tierney, "Digitized Voice-excited Vocoder for Telephone-quality Inputs, Using Bandpass Sampling of the Baseband Signals," J. Acoust. Soc. Amer., Vol. 37, pp. 753-754, April 1965.
5. C.K. Un and D.T. Magill, "the Residual-Excited Linear Prediction Vocoder with Transmission Rate Below 9.6 kb/s," IEEE Trans. Comm., Vol. COM-23, pp. 1466-1474, Dec. 1975.
6. C.J. Weinstein, "A Linear Prediction Vocoder with Voice Excitation," Proc. EASCON '75, pp. 30A-30G, Sept. 29-Oct. 1, 1975, Washington, D.C.
7. D. Esteban, C. Galand, D. Mauduit, and J. Menez, "9.6/7.2 KBPS Voice Excited Predictive Coder (VEPC)," Int. Conf. Acoustics, Speech, Signal Processing, Tulsa, OK, pp. 307-311, April 1978.
8. B.S. Atal, M.R. Schroeder, and V. Stover, "Voice-Excited Predictive Coding System for Low Bit-Rate Transmission of Speech," Int. Conf. Comm., June 16-18, 1975, San Francisco.

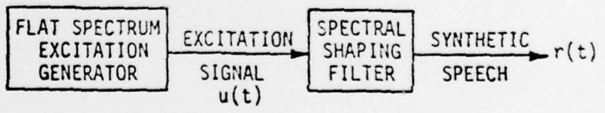


Fig. 1 Basic synthesis model for baseband coder.

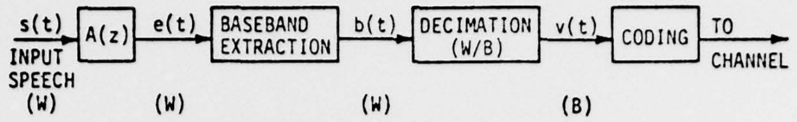


Fig. 2 Transmitter for a baseband coder.

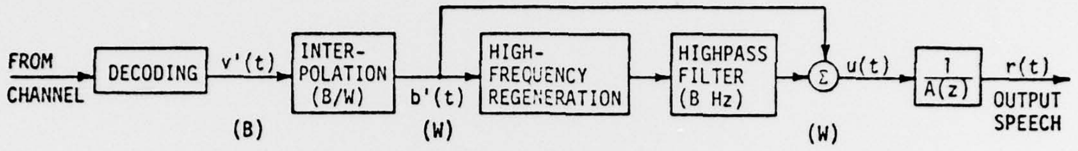


Fig. 3 Typical receiver for a baseband coder.

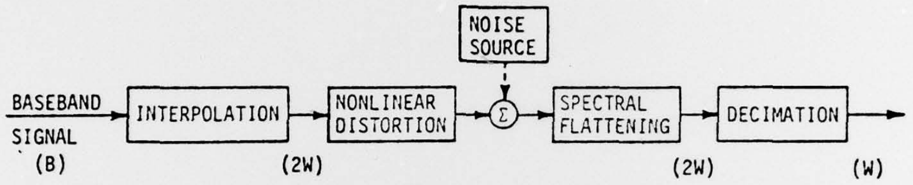


Fig. 4 High-frequency regeneration by nonlinear distortion and spectral flattening. The output signal in this figure is the output of the HFR block in Fig. 3.

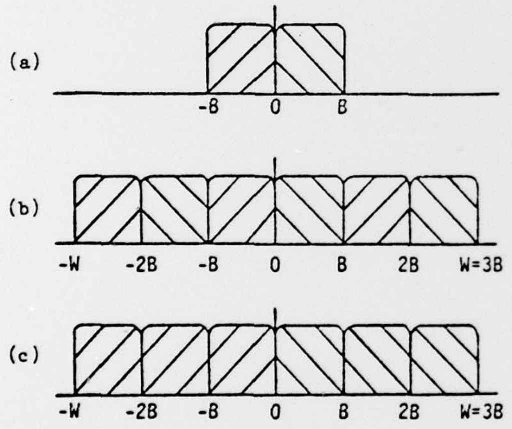


Fig. 5 (a) Baseband spectrum
(b) 3-band spectral folding
(c) 3-band spectral translation.

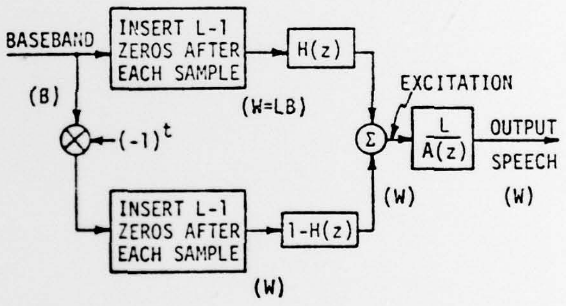


Fig. 7 Receiver for baseband coder that uses integer-band spectral translation.

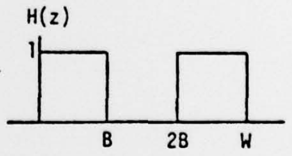


Fig. 8 The filter H(z) for W=3B.

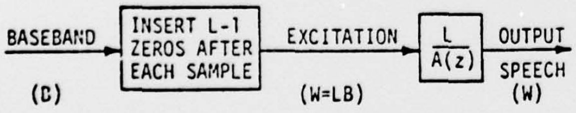


Fig. 6 Receiver for baseband coder that uses integer-band spectral folding.

APPENDIX D

DIPHONE SYNTHESIS FOR PHONETIC VOCODING

(Presentation given at the IEEE Int. Conf. Acoustics, Speech, and Signal Processing, Washington, DC, pp. 891-894, April 1979.)

IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Washington, DC, April 1979.

DIPHONE SYNTHESIS FOR PHONETIC VOCODING

R. Schwartz, J. Klovstad, J. Makhoul,
D. Klatt, and V. Zue

Bolt Beranek and Newman Inc.
Cambridge, Mass.

ABSTRACT

We report on the synthesis of speech in the context of a phonetic vocoder operating at 100 b/s. With each phoneme, the vocoder transmits the duration and a single pitch value. The synthesizer uses a large inventory of diphone "models" to synthesize a desired phoneme string. The diphone inventory has been selected to differentiate between prevocalic and postvocalic allophones of sonorants, to account for changes in vowel color conditioned by postvocalic liquids, to allow exact specification of voice onset time, and to permit synthesis of glottal stops, alveolar flaps and syllabic consonants. The diphones are extracted from carefully constructed short utterances and are stored as a sequence of LPC parameters. During synthesis, the requisite diphone models are time-warped, abutted and smoothed to produce a complete sequence of LPC parameters that are used in the synthesis. The algorithms used are described and compared with more conventional methods. Examples of the synthesized speech will be played.

1. INTRODUCTION

We have developed a phonetic speech synthesizer that is designed to operate as the synthesis part of a very-low-rate (less than 100 bits per second) speech transmission system [1]. It also has application as part of text-to-speech and voice response systems.

A string of phonemes to be synthesized is first translated into a corresponding diphone sequence. A diphone is defined as the region from the middle of one phoneme to the middle of the next phoneme. A diphone template consists of the LPC parameters necessary for synthesizing one diphone. The LPC parameters, which are assembled by concatenating diphone templates, are then used to synthesize speech.

Dixon and Maxey [2] used diphone segment assembly in a terminal analog speech synthesis system. There have been a number of recent efforts at diphone-LPC synthesis [3,4]. One important goal of these recent systems was to minimize the storage and complexity. For instance, the "dyads" used by Olive were each represented by only two sets of log-area-ratio (LAR) parameters. In addition, Olive used simple linear interpolation of LAR parameters for reconstructing parameter tracks and for connecting between successive dyads. Olive also avoided using all permutations of phonemes, thereby reducing the number of dyads needed.

The overriding goal in our effort is to produce natural-sounding speech of a quality comparable to that of unquantized LPC vocoded speech. Consequently, we have resisted compromising speech quality by imposing limits on either storage or complexity. For instance, whenever a diphone is significantly affected by context we use a context dependent diphone. We have retained, however, the requirement of being able to specify the input to the synthesizer using less than 100 bits per second.

The following sections of this paper describe our diphone synthesis method in greater detail. Where appropriate, we shall compare our method to existing methods for diphone/dyad synthesis.

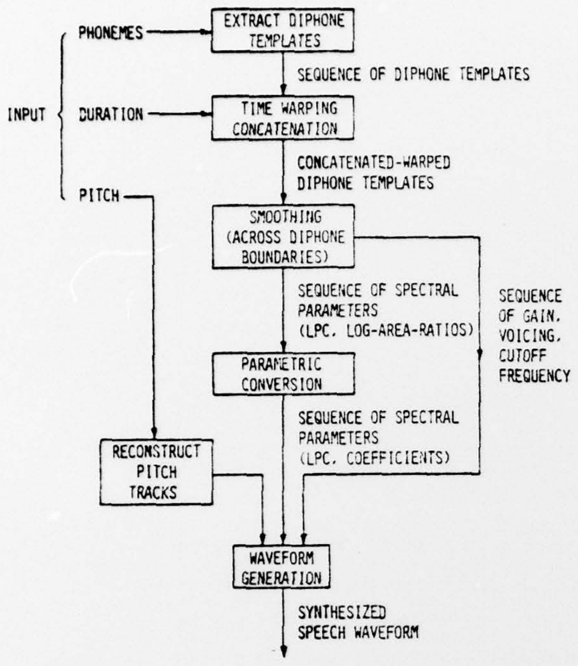
2. DIPHONE SYNTHESIS METHOD

The diphone is a natural unit for synthesis because the coarticulatory influence of one phoneme does not usually extend much further than half way into the next phoneme. Since diphone junctures are usually at articulatory steady states, minimal smoothing is required between adjacent diphones. Also the difficult task of duplicating phoneme transitions by complicated acoustic-phonetic rules is avoided. We estimate that approximately 2000 diphones are needed to achieve high quality.

A fundamental design decision in this research was to use diphone templates that have been extracted from real speech. We designed and recorded a set of nonsense utterances that contain all the possible diphones of English. The nonsense utterances in which these diphones were embedded were designed to provide a phonetically neutral context, in order that the diphone templates be applicable in a wide range of contexts. We also recorded those triphones (diphones in context) that were felt to be necessary. Whenever a diphone template extracted from a nonsense utterance is found to be inadequate, we extract a template from a more appropriate word or phrase.

Each template contains 14 LAR parameters and the gain for each frame of speech. The phoneme boundary (as indicated by manual transcription) is preserved.

Figure 1 illustrates the synthesis procedure. The speech synthesis program
1) expands the input phoneme sequence into a diphone sequence;
2) selects the most appropriate diphone template (depending on the local phonetic context);
3) time-warps each of the diphone templates to produce a gain track and 14 LAR parameter tracks of the specified durations;
4) smooths between consecutive warped diphone templates to minimize gain and spectral discontinuities;
5) reconstructs continuous pitch tracks by linear interpolation of the single pitch values given;



INPUT
PHONEMES
DURATION
PITCH
EXTRACT DIPHONE TEMPLATES
SEQUENCE OF DIPHONE TEMPLATES
TIME WARPING CONCATENATION
CONCATENATED-WARPED DIPHONE TEMPLATES
SMOOTHING (ACROSS DIPHONE BOUNDARIES)
SEQUENCE OF SPECTRAL PARAMETERS (LPC. LOG-AREA-RATIOS)
SEQUENCE OF GAIN, VOICING, CUTOFF FREQUENCY
PARAMETRIC CONVERSION
SEQUENCE OF SPECTRAL PARAMETERS (LPC. COEFFICIENTS)
RECONSTRUCT PITCH TRACKS
WAVEFORM GENERATION
SYNTHESIZED SPEECH WAVEFORM

Fig. 1 Diphone synthesis method.

6) determines the cutoff frequency and voicing using knowledge of the phoneme being synthesized;
7) uses the resulting sequence of LARs, pitch, gain, and cutoff frequency (specified every 10 ms) as input to control an LPC speech synthesizer.

3. SYNTHESIS ALGORITHMS

Steps 1 and 7 given above are relatively straightforward procedures. This section will discuss the procedures in steps 2 through 6 in greater detail.

Extension to Diphones

In this section, we describe some extensions to the fundamental definition of a diphone. These extensions allow "special cases" to be handled uniformly by the synthesis program (thus permitting the highest possible intelligibility and naturalness).

Context-Specific Diphones. Most diphones can be used adequately independent of context, but there are important exceptions. For instance, in the sequence [W-IH-L], as in the word "will", the part of the phoneme [IH] contained in the diphone [W-IH] is drastically affected by the presence of the [L]. Consequently, there must be a separate diphone template for [W-IH] to be used in this context. To account for such phenomena we have allowed more than one template to be defined for diphones when they are affected by context. Additional diphone templates necessitated by lateralization, retroflexion and other strong contextual phenomena are expected to account for 10-20% of the total diphone inventory.

Splitting Phonemes. Some phonemes that have more than one acoustically distinct region have been split up into two "pseudophonemes". The diphthongs, for example, have two very different acoustic regions. For instance, the diphthong [AY], as in "bite", starts out much like an [AA], as in "pot", but ends up more like the [IY] in "beet". The two relatively steady regions are connected by a rapid transition between them. The durations of the two steady regions are somewhat independent, as are the contextual effects of neighboring phonemes. Therefore, some of the diphthongs have been split into two pseudophonemes (e.g., [AY1,AY2]), which appear only in sequence. The unvoiced plosives and affricates also have two acoustically distinct regions. Each region is treated as if it were a separate phoneme.

Both the context-dependent diphone templates and the pseudophonemes described above are determinable from a normal phoneme string. The only change in data rate is due to the extra pseudophoneme duration (about 2 bits).

Time Warping

In order to provide input to the LPC synthesis program we must specify LPC coefficients at fixed intervals (10 ms) by time warping template information (whose duration is fixed) to satisfy phoneme durations specified by the input. This is made difficult by the fact that the time warping must preserve the naturalness of speech. One way to do this is to treat speech as being made up of elastic and inelastic regions.

The principle of distinguishing between "elastic" and "inelastic" regions of the template arises from observations of speech parameters under widely varying speaking rates. Most of the durational variation is observed to occur during the "steady state" portion of the phoneme (an elastic region), whereas the transition portions (inelastic regions) are relatively insensitive to changes in speaking rate. Hence, our time warping algorithm allows us to specify that a certain percentage of the diphone template on each side of the phoneme boundary is to be treated as relatively inelastic and the rest of the diphone template (the section corresponding to phoneme middles) as more elastic.

Time warping is accomplished by the use of piecewise linear mapping functions, which define that part of the template to be used at each instant of time. This correspondence and the resulting mapping function are illustrated in Figure 2. The speech being synthesized is the sequence /- DH AX M/ as in "The man...". The vertical axis represents time in the diphone templates. The horizontal axis represents time in the synthesized speech. The diphone template durations are determined from the original recorded nonsense utterances, while the phoneme durations are determined by the input utterance to be vocoded.

The phoneme boundaries in the diphone templates are mapped onto the phoneme boundaries in the synthesized speech to uniquely define a point in the piecewise linear mapping function. The diphone boundary is mapped onto the center of the phonemes. The DASHED lines connect phoneme boundaries, and the DASH-DOT lines connect diphone boundaries (phoneme centers). Notice that the templates are generally compressed during the mapping. The reason for this is that (whenever possible) our diphone templates are extracted from

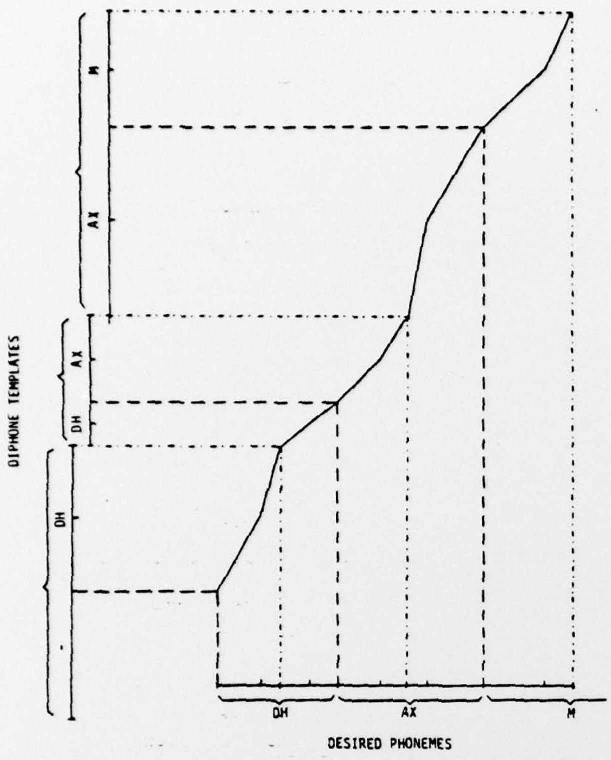


Fig. 2 Piecewise linear diphone mapping function.

fully articulated *short utterances. Furthermore*, we believe that compressing a long template will result in better speech quality than expanding a short one, since information is more easily ignored than generated.

The "elastic" and "inelastic" regions are delineated by small tic marks on both axes. The knees in the mapping function shown correspond to the intersection of these tic marks. *Although both* elastic and inelastic regions *of* the template (in this example) are shortened by mapping from templates to phonemes, the inelastic regions are shortened less.

As part of our test of the warping algorithm, we tried varying the rate of the synthesized speech. When the time warping was uniform, some phoneme transitions (e.g., vowel/nasal transitions) became less intelligible. The time warping relations were varied such that all the transitions sounded natural over a wide range of speech rates.

Parameter Continuity

When templates are concatenated, discontinuities in the gain and the LAR parameters may result despite the fact that the parameters are smooth within each template. In order to deal with this, we define an "interpolation region" that straddles the diphone template boundary (phoneme *middle) which* contains potential parametric discontinuities. This interpolation region is defined by two "interpolation points" (one from each diphone).

Figure 3 compares two different parameter smoothing algorithms as applied to a single parameter. The heavy line indicates the parameter

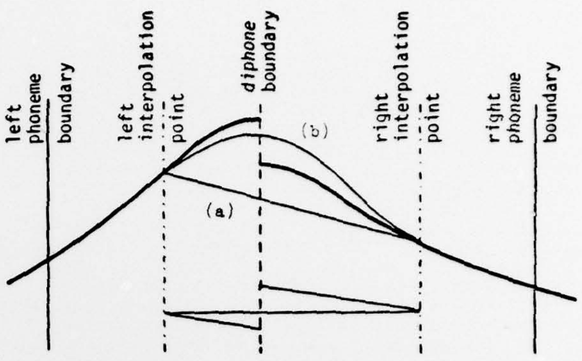


Fig. 3 Parameter Smoothing. (a) Linear interpolation, (b) Adding ramps with equal slope.

tracks as taken from the two diphone templates. Taken together these tracks span one phoneme. *There is a discontinuity at the* diphone boundary, which is indicated by the vertical DASHED line. The two vertical DASH-DOT lines delineate the interpolation region. The straight line (a) connecting the two *parameter* tracks illustrates the effect of linear interpolation. As can be seen in this case the linear interpolation results in a poor fit to the original data. The other curve (b) connecting the two points is derived by adding a ramp (shown at *the bottom of the figure*) to each parameter track, such that the discontinuity is eliminated. This second smoothing method often yields a better fit to the original parameter tracks than linear interpolation. Note that the second method requires that the parameter tracks be represented in the diphone templates by more than two frames per diphone. It is expected that 4 to 6 frames per diphone will be sufficient.

Excitation

In addition to 14 LAR parameters and gain, the LPC synthesis program requires for each 10 ms frame a value of pitch, a voicing flag, and a cutoff frequency.

Pitch Track. The pitch track during voiced phonemes is reconstructed from the input pitch values by straight line interpolation. In our simulation of the analysis part of a phonetic vocoder, the single transmitted pitch values are determined from complete pitch tracks in the sentence being analyzed. The analysis program (given the phoneme identities and phoneme boundaries) determines a weighted piecewise linear least squares fit to that pitch track. The endpoints of the linear sections (which occur at phoneme boundaries) are transmitted. The weighting is designed to minimize the effect of pitch tracker errors. It was found that sentences synthesized using these piecewise linear pitch tracks are practically indistinguishable from those using the original analyzed pitch tracks.

Voicing. Voicing was determined directly from the identity of the phoneme being synthesized. Voicing errors are avoided by careful placement of phoneme boundaries in the diphone templates. We have found that a one frame error in placement of the boundary can cause a severe "pop" in the synthesized speech, due to misalignment of spectral parameters with excitation parameters.

Mixed-Source Model - Cutoff Frequency. Previous work [5,6] has shown that our mixed-source model of excitation results in more natural sounding (less buzzy) speech by allowing for specification of a cutoff frequency with every value of pitch. The voicing excitation is low-pass filtered and the frication is high-pass filtered. The cutoff frequency simultaneously marks the upper edge of the voicing spectrum and the lower edge of the frication spectrum. Currently we have implemented an algorithm that selects a cutoff frequency based on distinctive features of the phoneme being synthesized. For example, for vowels the cutoff frequency is at 5000 Hz (fully voiced); for unvoiced consonants it is 0 Hz; and for voiced fricatives (which are produced with both periodic and random excitation) it is 1500 Hz. The cutoff frequency parameter track is then low-pass filtered in time in order to minimize excitation discontinuities at phoneme boundaries. The implementation of the cutoff-frequency algorithm has resulted in a marked improvement in speech quality: in particular, a decrease in buzziness.

4. RESULTS

Several sentences have been synthesized using the techniques described above. The resulting speech was highly intelligible. There are occasional instances where the intensity of a vowel is inappropriate. This is to be expected since vowel intensity varies with stress - which is not given as an input to the synthesizer. To alleviate this problem, a gain adjustment could be estimated from knowledge of the phoneme, its duration, and associated pitch value (in relation to those data for neighboring phonemes). Alternatively, in a speech transmission system, the analyzer could send a phoneme specific gain adjustment for each vowel using a small number of bits (probably one). (Since there are only about 5 vowels per second in normal speech, this represents a small increase in bit rate.) In a text-to-speech or synthesis-by-rule application, this gain adjustment is derived from syllable stress information.

The quality of unquantized LPC vocoded speech was judged to be somewhat better than that of synthesized speech; however, the differences were not large. Obviously the quality of the synthesized speech cannot be any better than that of vocoded speech. Nonetheless, our aim is to achieve that quality.

We tried synthesizing the speech of a second (male) speaker by using the phoneme duration and pitch values from the new speaker in conjunction with the diphone templates extracted from the speech of our model speaker. The synthesized speech had definite characteristics of both speakers. However, the characteristics of the new speaker (as conveyed in the pitch and duration values) were dominant.

5. DATA REDUCTION

During this initial phase of the research, we have preserved all of the template data without frame rate reduction or parameter quantization. We anticipate that we shall be able to achieve a data reduction comparable to that used in normal LPC vocoding. Allowing for a frame rate of 4 to 6 frames per diphone (twice the frame rate of variable rate LPC vocoding), we would project a total storage requirement of less than 400,000 bits. While this may be 2 or 3 times that required by simpler methods, we feel that, for many applications, the improved naturalness will be worth the added storage.

6. CONCLUSION

We have described a phonetic speech synthesis program that uses diphone template concatenation and LPC synthesis to produce highly intelligible speech of quality comparable to that of LPC vocoded speech. Every effort has been taken to preserve the information necessary and use algorithms sufficiently complex to maximize the naturalness of the synthesized speech.

REFERENCES

1. J. Makhoul, C. Cook, R. Schwartz, and D. Klatt, "A Feasibility Study of Very Low Rate Speech Compression Systems," Report No. 3508, Bolt Beranek and Newman Inc., Cambridge, Mass., Feb. 1977.
2. R. Dixon and H. Maxey, "Terminal Analog Synthesis of Continuous Speech Using the Diphone Method of Segment Assembly," IEEE Trans. on Audio and Electroacoustics, Vol. AU-16, No. 1, March 1968, pp. 40-50.
3. J. Olive, "Speech Synthesis from Phonemic Transcription," presented at the 96th meeting of the Acoustical Society of America, Nov. 1978, paper GGG24.
4. R. Gillman, "Some Experiments With an Area Function Dyad Synthesizer," presented at the 96th meeting of the Acoustical Society of America, Nov. 1978, paper GGG11.
5. J. Makhoul, R. Viswanathan, R. Schwartz, and A.W.F. Huggins, "A Mixed-Source Model for Speech Compression and Synthesis," 1978 IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Tulsa, Okla., April 10-12, 1978, pp. 163-166. (To appear in J. Acoust. Soc. Am., Dec. 1978.)
6. A.W.F. Huggins, R. Schwartz, R. Viswanathan and J. Makhoul, "Subjective Quality Testing of a New Source Model of LPC Vocoders," presented at the 96th meeting of the Acoustical Society of America, Nov. 1978, paper GGG12.

ACKNOWLEDGMENT

This work was supported by the Advanced Research Projects Agency and monitored by RADC/ETC under Contract No. F19628-78-C-0136.